НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА СИСТЕМНОГО ПРОГРАМУВАННЯ І
СПЕЦІАЛІЗОВАНИХ КОМП'ЮТЕРНИХ СИСТЕМ

«На правах рукопису»
УДК _____

«До захисту допущено»

Завідувач кафедри СПСКС

_____ В.П.Тарасенко_
(підпис) (ініціали, прізвище)

"___"_____2018р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 123 Комп'ютерна інженерія
Комп'ютерні системи та компоненти

на тему: Класифікація веб-сторінок на основі технології інтелектуального аналізу даних

Виконала: студентка VI курсу, групи КВ-61м

Кузьомка Марина Анатоліївна _____
(підпис)

Науковий керівник к.т.н., доцент Боярінова Ю. Є. _____
(підпис)

Рецензент_____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2018 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп'ютерних систем

Рівень вищої освіти – другий (магістерський)
Спеціальність 123 Комп'ютерна інженерія
Комп'ютерні системи та компоненти

ЗАТВЕРДЖУЮ
Завідувач кафедри СПСКС
_____ _В.П.Тарасенко__
(підпис) (ініціали, прізвище)

«___»_____2018р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Кузьомки Марини Анатоліївни

1. Тема дисертації Класифікація веб-сторінок на основі технології інтелектуального аналізу даних,
науковий керівник дисертації Боярінова Ю. Є., к.т.н., доцент,
затверджені наказом по університету від «22» березня 2018 р. №986-с

2. Термін подання студентом дисертації 11 травня 2018 р.

3. Об'єкт дослідження - методи і алгоритми інтелектуального аналізу даних. _

4. Предмет дослідження - методи аналізу і класифікації веб-сторінок.

5. Перелік завдань, які потрібно розробити:
   - дослідження існуючих методів та алгоритмів інтелектуального аналізу даних;
   - дослідження існуючих методик класифікації веб-контенту;
   - вибір і вивчення інструментів інтелектуального аналізу даних;
   - підвищення рівня точності методів класифікації веб-контенту.

6. Перелік ілюстративного матеріалу
   - - Порівняльна характеристика методів інтелектуального аналізу даних;

- Етапи інтелектуального аналізу даних;
- Етапи роботи з адресами веб-сайтів;
- Етапи фільтрації тексту;
- Результати векторизації;
- Порівняльний аналіз розроблюваних методик класифікації.

7. Перелік публікацій: Система для класифікація веб-сайтів на основі методів інтелектуального аналізу даних була представлена та обговорювалась на наукових конференціях магістрантів та аспірантів «Прикладна математика та комп'ютинг» ПМК-2018 (Київ, 21-23 березня 2018 р.) та «Теоретичні та прикладні аспекти розробки комп'ютерних систем '2018» (Київ, 29 – 30 березня 2018 р.).

8. Дата видачі завдання 5 вересня 2016 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Ознайомлення з предметною областю дослідження.	20.11.2016	
2	Визначення структури магістерської дисертації; вивчення літератури, пошук додаткової літератури.	15.02.2017	
3	Робота над першим розділом магістерської дисертації	30.05.2017	
4	Проведення наукового дослідження; робота над другим розділом магістерської дисертації; розроблення програмного забезпечення	31.01.2018	
5	Проведення наукового дослідження; робота над статтею за результатами наукового дослідження;	25.02.2018	
6	Проведення наукового дослідження; робота над третім розділом магістерської дисертації; підготовка матеріалів доповіді на конференції ПМК-2018	20.03.2018	
7	Підготовка графічної частини дипломного проекту	10.04.2018	
8	Оформлення документації дипломного проекту	22.04.2018	
9	Попередній розгляд магістерської дисертації на кафедрі	26.04.2018	

Студент _____ Кузьомка М.А.
(підпис)

Науковий керівник дисертації _____ Боярінова Ю.Є.
(підпис)

РЕФЕРАТ

Актуальність теми. Сьогодні Інтернет займає важливу роль в житті людини. Інформаційний простір в мережі налічує вже мільйони гігабайт даних різного роду і відрізняється високим рівнем доступності для користувачів. Для автоматизації перевірки і класифікації веб-контенту можна використовувати методи інтелектуального аналізу даних. Мета технології інтелектуального аналізу даних - виявити структури даних і знайти закономірності в слабоструктурованих даних. Інформація в Інтернеті відрізняється високою динамікою: створення нового контенту, його редагування та видалення займають кілька секунд. З огляду на кількість користувачів, які можуть створювати небажаний контент, використання традиційних методів виявлення та класифікації подібної інформації стає незручним.

Об'єктом дослідження є методи і алгоритми інтелектуального аналізу даних.

Предметом дослідження є методи аналізу і класифікації веб-сторінок.

Мета роботи: модифікація існуючих методів і алгоритмів інтелектуального аналізу даних для підвищення точності класифікації веб-сторінок.

Наукова новизна полягає в модифікації методів класифікації на основі «сусідніх» веб-сторінок, що дозволило підвищити точність класифікації.

Практична цінність:

1. В ході проведення аналізу досліджуваної області було визначено, що існуючі методи в повній мірі не задовольняють сучасним вимогам точності і повноти класифікації веб-сторінок.
2. Розроблені нові методи підвищення точності моделі класифікації веб-контенту на основі існуючих та розроблена модель, що дозволяє виконувати класифікацію веб-сторінок з точністю 96%.

Апробація роботи. Система для класифікація веб-сайтів на основі методів інтелектуального аналізу даних була представлена та обговорювалась на наукових конференціях магістрантів та аспірантів «Прикладна математика та комп'ютинг» ПМК-2018 (Київ, 21-23 березня 2018 р.) та «Теоретичні та прикладні аспекти розробки комп'ютерних систем '2018» (Київ, 29 – 30 березня 2018 р.).

Структура та обсяг роботи. Магістерська дисертація складається з вступу, трьох розділів та висновків.

У вступі подано загальну характеристику роботи, зроблено оцінку сучасного стану проблеми, обґрунтовано актуальність напрямку досліджень, сформульовано мету і задачі досліджень, показано наукову новизну отриманих результатів і практичну цінність роботи, наведено відомості про апробацію результатів і їхнє впровадження.

У першому розділі розглянуто існуючі методи та інструменти для фільтрації контенту та методи класифікації веб-сторінок.

У другому розділі розглянуто методи інтелектуального аналізу даних для фільтрації контенту.

У третьому розділі розглянуто методи підвищення точності класифікації веб-сторінок та наведені особливості реалізації розробленої моделі.

У висновках представлені результати проведеної роботи.

Робота представлена на 82 аркушах, містить посилання на список використаних літературних джерел.

Ключові слова: інтелектуальний аналіз даних, класифікація, веб-сторінка, веб-контент.

ESSAY

Actuality of theme. Today the Internet plays an important role in human life. The information space in the network already has millions of gigabytes various kinds of data and has a high level of accessibility for users. You can use the methods of data mining to automate the verification and classification of web content. The purpose of the technology of data mining is to detect data structures and to find patterns in poorly structured data. Information in the Internet is characterized by high dynamics: the creation of new content, its editing and deletion takes a few seconds. Given the number of users who can create unwanted content, using of the traditional methods for the detection and classification of such information becomes uncomfortable.

The object of the research is the methods and algorithms of data mining.

The subject of the research is the methods of analysis and classification of web pages.

Purpose of the research: the modification of the existing methods and algorithms of data mining to improve the accuracy of the classification of web pages.

The scientific novelty lies in modification classification methods based on " neighboring" web pages that allowed to improve the accuracy of the classification.

Practical value:

1. During the analysis of the studied area it was determined that the existing methods do not fully meet the current requirements for the accuracy and completeness of the classification of web pages.
2. There were developed new methods to improve the accuracy of the web content classification model based on the existing methods and was developed model, which allows to classify web pages with an accuracy of 96%.

Approbation of the work. The system for the classification of websites based on the methods of data mining was presented and discussed at the

scientific conferences of masters and postgraduates "Applied mathematics and computer science", PMK-2018 (Kyiv, March 21-23, 2018) and "Theoretical and applied aspects development of computer systems' 2018 "(Kiev, March 29-30, 2018).

Structure and scope of work. The master's dissertation consists of an introduction, three sections and conclusions.

The introduction gives a general description of the work, assesses the current state of the problem, substantiates the relevance of the research direction, formulates the purpose and objectives of the research, shows the scientific novelty of the obtained results and the practical value of the work, provides information about the approbation of the results and their implementation.

The first section discusses existing methods and tools for content filtering and web page classification methods.

The second section discusses methods of data mining for content filtering.

The third section presentes the methods of increasing the accuracy of the classification of web pages and features of implementation of the developed model.

The conclusions shows the results of the work.

The work is presented on 82 sheets, contains a link to the list of used literary sources.

Keywords: data mining, classification, web page, web content.

РЕФЕРАТ

Актуальность темы. Сегодня Интернет занимает важную роль в жизни человека. Информационное пространство в сети насчитывает уже миллионы гигабайт данных различного рода и отличается высоким уровнем доступности для пользователей. Для автоматизации проверки и классификации веб-контента можно использовать методы интеллектуального анализа данных. Цель технологии интеллектуального анализа данных - выявить структуры данных и найти закономерности в слабоструктурированных данных. Информация в Интернете отличается высокой динамикой: создание нового контента, его редактирование и удаление занимают несколько секунд. Учитывая количество пользователей, которые могут создавать нежелательный контент, использование традиционных методов выявления и классификации подобной информации становится неудобным.

Объектом исследования являются методы и алгоритмы интеллектуального анализа данных.

Предметом исследования являются методы анализа и классификации веб-страниц.

Цель работы: модификация существующих методов и алгоритмов интеллектуального анализа данных для повышения точности классификации веб-страниц.

Научная новизна заключается в модификации методов классификации на основе «соседних» веб-страниц, что позволило повысить точность классификации.

Практическая ценность:

1. В ходе проведения анализа исследуемой области было определено, что существующие методы в полной мере не удовлетворяют современным требованиям точности и полноты классификации веб-страниц.
2. Разработаны новые методы повышения точности модели

классификации веб-контента на основе существующих и разработана модель, позволяющая выполнять классификацию веб-страниц с точностью 96%.

Апробация работы. Система для классификация веб-сайтов на основе методов интеллектуального анализа данных была представлена и обсуждалась на научных конференциях магистрантов и аспирантов «Прикладная математика и компьютинг» ПМК-2018 (Киев, 21-23 марта 2018) и «Теоретические и прикладные аспекты разработки компьютерных систем '2018" (Киев, 29 - 30 марта 2018).

Структура и объем работы. Магистерская диссертация состоит из введения, трех глав и выводов.

Во *введении* представлена общая характеристика работы, произведена оценка современного состояния проблемы, обоснована актуальность направления исследований, сформулированы цели и задачи исследований, показано научную новизну полученных результатов и практическую ценность работы, приведены сведения об апробации результатов и их внедрение.

В *первом разделе* рассмотрены существующие методы и инструменты для фильтрации контента и методы классификации веб-страниц.

Во *втором разделе* рассмотрены методы интеллектуального анализа данных для фильтрации контента.

В *третьем разделе* рассмотрены методы повышения точности классификации веб-страниц и приведены особенности реализации разработанной модели.

В *выводах* представлены результаты проведенной работы.

Работа представлена на 82 страницах, содержит ссылки на список использованных литературных источников.

Ключевые слова: интеллектуальный анализ данных, классификация, веб-страница, веб-контент.

ЗМІСТ

ДОДАТКИ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

Bagging - технологія класифікації, коли всі елементарні класифікатори навчаються і працюють паралельно (незалежно один від одного);

CRISP-DM - Cross-Industry Standard for Data Mining - модель, що описує основні етапи, виконання яких дозволяє організаціям отримувати максимальну вигоду від використання методів інтелектуального аналізу даних;

Cross-validation - метод оцінювання достовірності математичної моделі з метою перевірки, наскільки результати статистичного аналізу узагальнюються на незалежному наборі даних;

Data Mining - виявлення прихованих закономірностей або взаємозв'язків між змінними у великих масивах необроблених даних;

Decision tree - засіб підтримки прийняття рішень, що використовується в статистиці і аналізі даних для прогнозних моделей;

JDBC - Java DataBase Connectivity - прикладний програмний інтерфейс Java, який визначає методи, з допомогою яких програмне забезпечення на Java здійснює доступ до бази даних.

KDD - Knowledge Discovery in Databases - процес пошуку корисних знань в "сирих" даних;

kNN - k-Nearest Neighbors - найпростіший метричний класифікатор, що заснований на оцінюванні подібності об'єктів. Об'єкт, що класифікується, відноситься до того класу, якому належать найближчі до нього об'єкти навчальної вибірки;

N-gram - послідовність з n елементів, це може бути послідовність звуків, складів, слів або літер;

Random Forest - алгоритм машинного навчання, що полягає у використанні комітету (ансамблю) вирішальних дерев;

ROC - графік, що дозволяє оцінити якість бінарної класифікації, відображає співвідношення між часткою об'єктів від загальної кількості носіїв ознаки, вірно класифікованих від загальної кількості об'єктів, що не несуть ознаки, помилково класифікованих, як мають ознаку.

SVM - Support Vector Machine - набір схожих алгоритмів навчання з учителем, що використовуються для задач класифікації та регресійного аналізу.

TF-IDF - Term Frequency - Inverse Document Frequency - статистичний показник, що використовується для оцінки важливості слів у контексті документа, що є частиною колекції документів чи корпусу.

URL - Uniform Resource Locator - стандартизований спосіб запису адреси ресурсу в мережі.

ВСТУП

Сьогодні Інтернет займає важливу роль в житті людини. Інформаційний простір в мережі налічує вже мільйони гігабайт даних різного роду і відрізняється високим рівнем доступності для користувачів.

Легкість створення та редагування контенту в Інтернеті призводить до поширення небажаної інформації, зокрема забороненого контенту.

Інформація в Інтернеті відрізняється високою динамікою: створення нового контенту, його редагування та видалення займають кілька секунд. З огляду на кількість користувачів, які можуть створювати небажаний контент, використання традиційних методів виявлення та класифікації подібної інформації стає незручним.

Визначення тематики контенту веб-сторінок є однією з найважливіших задач багатьох інтернет-компаній. Наприклад, за умови коректної класифікації можна пропонувати користувачеві більш точну підбірку рекламних блоків, що в свою чергу дозволить підвищити продаж як місць розміщення рекламних банерів, так і рекламованого товару. Крім того, захист від небажаної інформації також є однією з основних можливих сфер застосування класифікації контенту.

Для автоматизації перевірки і класифікації веб-контенту, а також для виявлення небажаних для перегляду веб-сторінок і веб-сайтів, можна використати методи інтелектуального аналізу даних. Завдання технології інтелектуального аналізу даних - виявити структури даних і знайти закономірності в слабоструктурованих даних. Зважаючи на точність класифікації, що дають існуючі методи, можна зробити висновок, що такі методи потребують модифікації.

Метою роботи є дослідження способів класифікації веб-сторінок за допомогою існуючих моделей, методів і алгоритмів інтелектуального аналізу даних, модифікація цих методів та підвищення їх точності.

Основними завданнями роботи є:

- дослідження існуючих методів та алгоритмів інтелектуального аналізу даних;
- дослідження існуючих методик класифікації веб-контенту;
- вибір і вивчення інструментів інтелектуального аналізу даних;
- підвищення рівня точності методів класифікації веб-контенту.

1. АНАЛІЗ ІНСУЮЧИХ МЕТОДІВ ФІЛЬТРАЦІЇ КОНТЕНТУ

1.1. Методи фільтрації контенту

Під фільтрацією контенту мається на увазі програмне забезпечення, яке дозволяє обмежити доступ до небажаного контенту в мережі для певного кола людей.

Найчастіше фільтрація контенту відбувається на рівні веб-запитів протоколу HTTP. В такому випадку URL веб-сайту порівнюється з «чорним» списком, для такого порівняння зазвичай використовуються регулярні вирази.

«Чорні» списки потрібно часто оновлювати, адже в іншому випадку захист з їх допомогою стає малоефективним. Найбільш якісними є методи класифікації і обробки природної мови. В такому випадку класифікація веб-сайтів виконується за допомогою аналізатора кількості ключових слів за різними ознаками. Властивості, що отримуються з тексту, використовуються для визначення ступеня ймовірності відповідності небажаним категоріям. У випадку, коли ймовірність стає вище встановлених значень, відбувається блокування доступу.

Найпростіші програми дозволяють ввести слова, а система буде вести їх пошук вручну. В той же час більш складні програми мають великий словник і мають вже готову базу посилань, що були попередньо класифіковані. Як правило, розробники забезпечують періодичне оновлення бази посилань більш складних програм. Якщо веб-сайт не класифікований автоматично, то людина переглядає його і привласнює категорію сайту вручну.

Зрозуміло, що швидкодія класифікації – одна з найбільш важливих вимог до програм обмеження доступу.

Фільтрація контенту – це обмеження доступу користувачів до веб-сторінок. Головними методами аналізу контенту вважаються системи

тематичної класифікації вмісту веб-сторінки та пошук за ключовими словами (стоп-словами) [1].

1.1.1. Динамічне визначення тематичної категорії

Для визначення категорії вміст сторінки аналізується з використанням обробки природної мови при зверненні користувача до неї. Після чого виконується встановлення категорії або декількох категорій за результатами аналізу.

Є два види такої фільтрації:

- фільтрація вмісту запитів;
- фільтрація вмісту сайтів.

Перевагою даного способу є те, що створюється модель, яка в онлайн режимі визначає, чи може користувач переглядати даний сайт чи ні. Складність цього методу полягає в створенні досконалої моделі розпізнавання вмісту та в подальшій класифікації конкретного сайту.

Застосування цієї технології надає ряд істотних переваг:

- Проведений аналіз всіх веб-сторінок знижує шанс доступу до небажаного контенту, адже існує ряд сайтів з динамічною генерацією адрес.
- Зникає потреба постійного оновлення списків, тому що проводиться постійний аналіз тексту, незалежно від того, коли цей текст змінювався.
- Блокування виконується на рівні однієї веб-сторінки, а не всього веб-сайту, що дозволяє вирішити проблеми, пов'язані з веб-сайтами складної класифікації, такі як новини, наприклад [1].

1.1.2. Списки URL

Найбільш популярною технологією фільтрації контенту є «чорні» списки. В основі цього лежить створення та наповнення списку адрес веб-

сайтів, що вже були класифіковані.

Системи фільтрації веб-контенту з використанням списків URL можуть використовувати як локальні сховища, так і віддалені бази даних.

Коли використовується віддалена база даних, система відправляє запит на доступ до веб-сайту, де виконується пошук в базі даних і приймається рішення щодо надання доступу.

Локальні системи періодично оновлюють бази даних, тому для ефективної роботи необхідно оновлювати їх якомога частіше.

Фільтрація за списками має низку істотних недоліків:

- Повнота охоплення - списки URL не можуть містити всі можливі адреси всіх існуючих ресурсів Інтернету.
- Періодичність оновлення списків - контент веб-сайтів постійно змінюється, відбувається поява нових веб-сайтів і міграція старих, що в свою чергу накладає відбиток на якість такого підходу до класифікації.
- Соціальні мережі та блоги - основною особливістю подібних веб-сайтів є величезна кількість веб-контенту, який постійно змінюється та може бути небажаним для перегляду [2].

1.1.3. Фільтрація за ключовими словами

Метод фільтрації за ключовими словами полягає у пошуку в тексті певних ключових слів або словосполучень. Якщо веб-сторінка містить подібні слова і словосполучення, то відбувається блокування доступу до цієї веб-сторінки.

Описаний метод дозволяє відмінно фільтрувати контент за умови, коли наявність певних словосполучень або слів може однозначно визначити ступінь небезпеки для подальшого блокування веб-сторінки. В даному випадку це означає, що при вживанні таких слів контекст грати ролі не повинен.

Найчастіше, коли веб-контент перевіряється на рахунок вмісту ключових слів, без аналізу контексту неможливо однозначно відповісти на питання, чи варто виконувати блокування веб-сайту чи ні.

Тому даний спосіб перевірки доцільніше використовувати як доповнення при використанні інших технологій фільтрації.

1.2. Існуючі методики веб-класифікації

Сьогодні існує безліч наукових робіт в області класифікації веб-сторінок. Основною відмінністю класифікації веб-сторінок від звичайного тексту є гіпертекст. Зважаючи на це, можна виділити два типи класифікації:

- класифікація за вмістом цільової веб-сторінці;
- класифікація за вмістом сусідніх веб-сторінок.

Розглянемо можливі існуючі методи класифікації.

Серед атрибутів веб-сторінок розрізняють текстові та візуальні. Для класифікації текстова інформація - зручніша для використання. Для цього використовуються кілька варіантів вибору атрибутів, такі як bag-of-words, TF-IDF і n-gram. Такі методи зазвичай застосовуються в дослідженнях аналізу тексту.

Веб-сторінка використовує HTML теги, які застосовуються в якості контейнерів, в яких може знаходитись текст. Такі теги можуть бути обрані в якості атрибутів.

Веб-сторінку можна представити у вигляді ієрархії візуальних елементів, таких як навігація, контент та інші блоки. Не завжди впровадження такого методу дозволяє збільшити точність. В результаті можна об'єднати дані підходи для підвищення точності класифікації. Продуктивність моделі класифікації можна також поліпшити за рахунок зменшення розмірності даних.

Використання сусідніх веб-сторінок дозволяє значно підвищити точність. Найбільш корисними для класифікації є веб-сторінки, на які посилаються батьківська веб-сторінка цільової, а також веб-сторінки з тими ж посиланнями. При цьому сусідні веб-сторінки також можуть додавати велику кількість шуму. Якщо розглядати зв'язки між веб-сторінками, то можна побудувати граф і на основі цього отримати вектор, використовуючи методику як в TF-IDF.

Кожна веб-сторінка має свою унікальну адресу URL, за допомогою якої можна виконати порівняно швидку класифікацію без скачування веб-сторінки. Використання n-gram при класифікації за URL веб-сторінки також може підвищити ефективність [3].

1.3. Постановка задачі подальших досліджень

Метою роботи є дослідження способів класифікації веб-сторінок за допомогою існуючих методів інтелектуального аналізу даних, модифікація цих методів, підвищення їх точності та розробка моделі, що дозволяє виконувати мультикласову класифікацію веб-сторінок.

В ході досліджень повинні бути проаналізовані існуючі методи та алгоритмів інтелектуального аналізу даних, методики класифікації веб-контенту. Базуючись на цьому, потрібно вибрати інструменти інтелектуального аналізу даних для підвищення рівня точності методів класифікації веб-контенту.

2. МЕТОДИ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ ДЛЯ ФІЛЬТРАЦІЇ КОНТЕНТУ

2.1. Інтелектуальний аналіз даних

Вперше поняття інтелектуального аналізу даних прозвучало в 1978 році. Спочатку його в основному застосовували для завдань обробки та аналізу даних в межах прикладної статистики. Також при цьому вирішувалися завдання обробки невеликих баз даних. В подальшому основним завданням інтелектуального аналізу даних став пошук прихованих знань у великих базах даних [4].

Інтелектуальний аналіз даних можна вважати міждисциплінарним напрямком, адже він охоплює такі галузі комп'ютерних наук як генетичні алгоритми, теорію нейронних мереж та еволюційне програмування. Методами інтелектуального аналізу даних також є і різні методи статистики:

- кореляційний аналіз;
- регресійний аналіз;
- дисперсійний аналіз;
- факторний аналіз;
- і т.д.

Важливу роль відіграють також методи машинного навчання та візуалізації результатів.

Так як після застосування інтелектуального аналізу даних можуть з’являтися не тільки корисні результати, а й хибні, то в такому випадку можна користуватися методами машинного навчання. За визначенням, машинне навчання - це наука, що вивчає комп'ютерні алгоритми, які автоматично покращуються під час роботи.

Візуалізація використовується для наочного представлення виявлених закономірностей. Застосування візуалізації дозволяє створити

графічний образ аналізованих даних, а сам образ допомагає побачити в процесі аналізу аномалії, структури і тренди. Для вирішення завдання візуалізації використовуються графічні методи, що показують наявність закономірностей в даних. Візуалізацію також можна розглядати як самостійний метод аналізу, що називається Visual Mining.

В процесі інтелектуального аналізу даних проводиться дослідження безлічі об'єктів. У більшості випадків його можна представити у вигляді таблиці, кожен рядок якої відповідає одному з варіантів, а в стовпці розміщені значення параметрів, що його характеризують. Залежна змінна - параметр, значення якого розглядається як залежне від інших параметрів. Власне, цю залежність і необхідно визначити, використовуючи методи інтелектуального аналізу даних.

Використання інтелектуального аналізу даних може включати два або три етапи:

- Виявлення закономірностей. На цьому етапі проводиться дослідження набору даних з метою пошуку прихованих закономірностей. Попередні гіпотези щодо виду закономірностей на даному етапі ще не формуються.

  На цьому етапі повинна здійснюватися перевірка достовірності знайдених закономірностей з використанням даних, що не брали участь в їх формуванні (мається на увазі контрольна вибірка). Результат перевірки при цьому повинен співпасти з попереднім результатом.

  Такий поділ даних на навчальну множину та множину, що перевіряється, часто використовують в методах нейронних мереж.
- Використання виявлених закономірностей для передбачення невідомих значень (прогнозоване моделювання). В випадку прогнозованого моделювання використовуються результати роботи першого етапу.

Прогнозоване моделювання включає такі етапи:

- прогноз невідомих значень;
- прогноз розвитку процесів.

Спільні закономірності, що шукають на першому етапі, формуються індуктивно, від часткового до загального. У результаті ми можемо отримати знання про деякий клас об'єктів на підставі дослідження сукупності окремих представників цього класу. Прогнозоване моделювання, навпаки, дедуктивне. Закономірності, що отримують на цьому етапі, формуються від загального до часткового.

- Аналіз винятків. Призначення даного етапу - виявлення і пояснення аномалій, що були попередньо знайдені в закономірностях. Можливо два варіанти:
  - Існує деяке логічне пояснення, яке також можна оформити у вигляді правила.
  - Відхилення трактується як помилка вихідних даних. В цьому випадку етап аналізу винятків можна використовувати для очищення даних [4].

Розглянемо декілька відомих класифікацій методів інтелектуального аналізу даних за різними ознаками.

Всі методи інтелектуального аналізу даних можна розділити на дві великі групи за принципом роботи з вихідними навчальними даними. У цій класифікації верхній рівень визначається на підставі того, чи зберігаються дані після інтелектуального аналізу даних або вони дистилюються для подальшого використання.

Відповідно маємо дві групи:

1. Безпосереднє використання даних, або збереження даних.

   У цьому випадку вихідні дані зберігаються в явному детальному вигляді і безпосередньо використовуються на стадіях

прогнозованого моделювання та/або аналізу винятків.

Недолік цієї групи методів в тому, що при їх використанні можуть виникнути складності аналізу надвеликих баз даних.

Методи цієї групи:

- кластерний аналіз;
- міркування за аналогією;
- метод найближчого сусіда;
- метод k-найближчого сусіда.

2. Виявлення і використання формалізованих закономірностей, або дистиляція шаблонів.

   За умови використання технології дистиляції шаблонів один зразок (шаблон) інформації витягується з вихідних даних і перетворюється в певні формальні конструкції, вид яких залежить від методу інтелектуального аналізу даних, що використовується.

   Цей процес виконується на стадії вільного пошуку, у першій же групі методів дана стадія в принципі відсутня. На стадіях прогнозованого моделювання та аналізу винятків використовуються результати стадії вільного пошуку, вони значно компактніше самих баз даних.

   Конструкції цих моделей можуть трактуватися аналітиком або ні ("чорні ящики").

   Методи цієї групи:

   - логічні методи;
   - методи кростабуляції;
   - методи візуалізації;
   - методи, засновані на рівняннях.

   Логічні методи, або методи логічної індукції, включають:

   - нечіткі запити і аналізи;
   - дерева рішень;

- символьні правила;
- генетичні алгоритми.

Методи цієї групи, мабуть, найбільше інтерпретуються, адже вони оформляють знайдені закономірності в досить прозорому вигляді з точки зору користувача. Отримані правила можуть включати безперервні і дискретні змінні. Слід зауважити, що дерева рішень можуть бути легко перетворені в набори символьних правил шляхом генерації одного правила по шляху від кореня дерева до його термінальної вершини. Дерева рішень і правила фактично є різними способами вирішення однієї задачі і відрізняються лише за своїми можливостями. Крім того, реалізація правил здійснюється більш повільними алгоритмами, ніж індукція дерев рішень.

Методи крос-табуляції включають:

- агенти;
- Баєсові мережі;
- крос-табличну візуалізацію.

Методи на основі рівнянь висловлюють виявлені закономірності у вигляді математичних виразів - рівнянь. Отже, вони можуть працювати лише з чисельними змінними, і змінні інших типів повинні бути закодовані відповідним чином. Це дещо обмежує застосування методів даної групи, проте вони широко використовуються при вирішенні різних завдань, особливо завдань прогнозування.

Основні методи даної групи: статистичні методи та нейронні мережі.

Статистичні методи найбільш часто застосовуються для вирішення завдань прогнозування. Існує безліч методів статистичного аналізу даних, наприклад:

- гармонійний аналіз.
- кореляційно-регресійний аналіз;
- кореляція рядів динаміки;
- виявлення тенденцій динамічних рядів;

Інша класифікація розділяє методи інтелектуального аналізу даних на дві групи:

- статистичні методи, що базуються на використанні усередненого досвіду, що відображається в даних, які накопичуються в БД за тривалий період;
- кібернетичні методи, що включають безліч різних математичних підходів.

Статистичні методи представляють собою чотири взаємопов’язані розділи:

- Попередній аналіз природи статистичних даних - полягає в перевірці гіпотез стаціонарності, незалежності, нормальності, однорідності, а також оцінці виду функції розподілу, її параметрів і т.д.
- Виявлення зв'язків та закономірностей - кореляційний аналіз, лінійний та нелінійний регресійний аналіз і т.д.
- Багатовимірний статистичний аналіз - кластерний аналіз, компонентний аналіз, лінійний та нелінійний дискрімінантний аналіз і т.д.
- Динамічні моделі та прогноз на основі часових рядів.

Кібернетичні методи інтелектуального аналізу даних - це набір підходів, що об'єднані ідеєю комп'ютерної математики та використанням теорії штучного інтелекту. До цієї групи належать такі методи:

- еволюційне програмування;
- штучні нейронні мережі (розпізнавання, кластеризація);
- генетичні алгоритми (оптимізація);
- асоціативні правила (пошук аналогів, прототипів);
- дерева рішень;
- нечітка логіка;
- системи обробки експертних знань.

Методи інтелектуального аналізу даних також можна класифікувати

за завданнями інтелектуального аналізу даних.

Відповідно до такої класифікації можна виділити дві групи. Перша група – це підрозділ методів, що займається вирішенням завдань сегментації (тобто завдання класифікації і кластеризації) та прогнозування.

Відповідно до другої класифікації методи інтелектуального аналізу даних можуть бути спрямовані на отримання описових і прогнозованих результатів.

Описові методи служать для знаходження шаблонів або зразків, що описують дані, які піддаються інтерпретації з точки зору аналітика.

До методів, спрямованих на отримання описових результатів, відносяться ітеративні методи кластерного аналізу, в тому числі:

- алгоритм k-медіани;
- алгоритм k-середніх;
- методи крос-табличної візуалізації;
- ієрархічні методи кластерного аналізу,
- карти Кохонена, що само організуються;
- різні методи візуалізації.

Прогнозовані методи використовують значення одних змінних для передбачення та прогнозування невідомих (пропущених) або майбутніх значень інших змінних.

До методів, що спрямовані на отримання прогнозованих результатів, відносяться такі методи:

- дерева рішень;
- лінійна регресія;
- нейронні мережі;
- метод найближчого сусіда;
- метод опорних векторів
- та ін.

Різні методи інтелектуального аналізу даних характеризуються

певними властивостями, які можуть бути визначальними при виборі методу аналізу даних. Методи можна порівнювати між собою, оцінюючи характеристики їх властивостей.

Серед основних властивостей і характеристик методів інтелектуального аналізу даних розглянемо наступні:

- масштабованість;
- точність;
- гнучкість;
- швидкість;
- інтерпретованість;
- трудомісткість;
- популярність.

На плакаті №1 представлена порівняльна характеристика деяких поширених методів інтелектуального аналізу даних. Оцінка кожної з характеристик проведена за наступними категоріями (в порядку зростання):

- надзвичайно низька;
- дуже низька;
- низька/нейтральна;
- нейтральна/низька;
- нейтральна;
- нейтральна/висока;
- висока;
- дуже висока.

Проаналізувавши наведену порівняльну характеристику методів інтелектуального аналізу даних можна зробити висновок, що кожен з методів має свої сильні і слабкі сторони. Але жоден метод, якою б не була його оцінка з точки зору властивих йому характеристик, не може забезпечити вирішення всього спектру завдань інтелектуального аналізу

даних.

Більшість інструментів інтелектуального аналізу даних, що пропонує зараз ринок програмного забезпечення, реалізують відразу кілька методів, наприклад, дерева рішень, індукцію правил та візуалізацію, або ж нейронні мережі, карти Кохонена, що само організуються, та візуалізацію.

В універсальних прикладних статистичних пакетах (наприклад, SPSS, SAS, STATGRAPHICS, Statistica, ін.) реалізується широкий спектр найрізноманітніших методів (як статистичних, так і кібернетичних). Слід враховувати, що для можливості їх використання, а також для інтерпретації результатів роботи статистичних методів (кореляційного, регресійного, факторного, дисперсійного аналізу і ін.) потрібні спеціальні знання в галузі статистики.

Універсальність того чи іншого інструменту часто накладає певні обмеження на його можливості. Перевагою використання таких універсальних пакетів є можливість відносно легко порівнювати результати побудованих моделей, отриманих різними методами. Така можливість реалізована, наприклад, в пакеті Statistica, де порівняння засноване на так званій "конкурентній оцінці моделей". Ця оцінка полягає в застосуванні різних моделей до одного і того ж набору даних і наступному порівнянні їх характеристик для вибору найкращої з них [4].

2.2. Завдання інтелектуального аналізу даних

В основу технології інтелектуального аналізу даних покладена концепція шаблонів, що представляють собою закономірності. В результаті виявлення цих закономірностей вирішуються завдання інтелектуального аналізу даних.

Завдання інтелектуального аналізу даних в залежності від способу їх вирішення можна розділити на два класи:

- навчання з учителем;
- навчання без вчителя.

У першому випадку потрібен навчальний набір даних, на якому створюється і навчається модель інтелектуального аналізу даних. В подальшому готова модель тестується і згодом використовується для передбачення значень в нових наборах даних.

У другому випадку мета завдань полягає у виявленні закономірностей, що наявні в існуючому наборі даних. Варто зауважити, що при цьому навчальна вибірка не потрібна.

В якості прикладу можна навести завдання аналізу споживчого кошика, коли в ході дослідження виявляються товари, що покупці найчастіше купують разом. До цього ж класу належить задача кластеризації.

Якщо говорити про класифікацію завдань інтелектуального аналізу даних за призначенням, то відповідно до неї, вони діляться на:

- описові;
- передбачливі.

Мета вирішення описових завдань - краще зрозуміти дані, що досліджуються, виявити наявні в них закономірності, навіть якщо в інших наборах даних вони не зустрічатимуться.

Для передбачливих завдань характерним є те, що в ході їх вирішення на підставі набору даних з відомими результатами будується модель для передбачення нових значень.

Основними завданнями інтелектуального аналізу даних є:

- класифікація;
- регресія;
- кластеризація;
- пошук асоціативних правил;
- пошук послідовності;

- прогнозування [4].

Розглянемо завдання класифікації.

В результаті вирішення цієї задачі можна виявити ознаки, які характеризують групи об'єктів досліджуваного набору даних, тобто, класи. Новий об'єкт за цими ознаками можна віднести до того чи іншого класу. Варто зазначити, що в цьому завданні більшість класів, до яких може бути віднесений об'єкт, відомі заздалегідь.

В інтелектуальному аналізі даних завдання класифікації по суті представляє собою завдання визначення значення одного з параметрів об'єкта, що аналізується, на підставі значень інших параметрів. Параметр, що аналізується, часто називають залежною змінною, а параметри, які беруть участь в його визначенні, - незалежними змінними. На рисунку 2.1 наведено приклад класифікації об'єктів у найпростішому випадку двох параметрів. Моделлю при цьому може служити дерево рішень:

if y <1.5

and y <2x - 2 then White

else Black

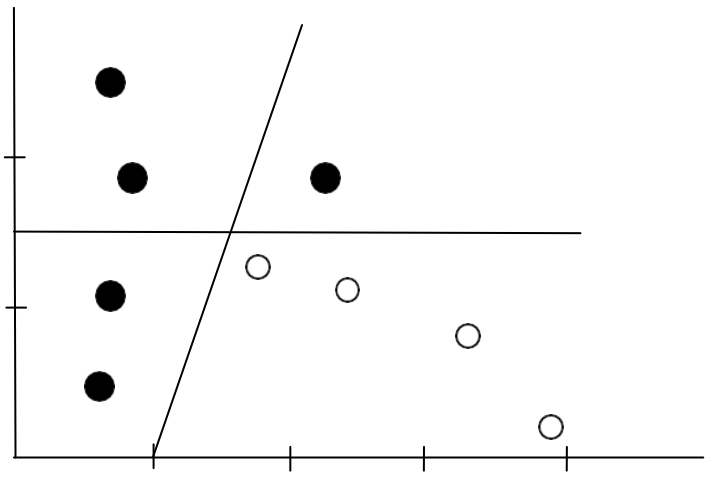


Рисунок 2.1 - Приклад класифікації

Аналогічний, хоча і більш складний, підхід використовуються для

вирішення практичних завдань класифікації. Наприклад, якщо людина звертається в банк, щоб отримати кредит, банківський службовець повинен прийняти рішення: чи кредитоспроможний потенційний клієнт чи ні. Таке рішення приймається на підставі даних про досліджуваний об'єкт (в даному випадку - про людину): місце його роботи, розмір заробітної плати, вік, склад сім'ї і т. д. В результаті аналізу цієї інформації банківський службовець повинен віднести людину до одного з двох відомих класів: «кредитоспроможний» або «некредитоспроможний». У розглянутому прикладі місце роботи, розмір зарабітної плати, вік, склад сім'ї - незалежні змінні, а кредитоспроможність клієнта - залежна змінна.

Іншим прикладом завдання класифікації є фільтрація електронної пошти. У цьому випадку програма фільтрації повинна класифікувати вхідне повідомлення як спам (небажану електронну пошта) або як лист. Дане рішення приймається на підставі частоти появи в спілкуванні певних слів (наприклад, імені одержувача, безособового звернення, слів і словосполучень: «придбати», «вигідна пропозиція» і т. д.).

У загальному випадку кількість класів в задачах класифікації може бути довільною. Наприклад, в задачі розпізнавання символів таких класів може бути стільки, скільки є символів. У такому завданні об'єктом класифікації є матриця пікселів, що представляє видимий образ символу, що розпізнається.

Завдання класифікації вирішується в два етапи. На першому виділяється навчальна вибірка. У неї входять об'єкти, для яких відомі значення як незалежних, так і залежних змінних. Для нашого прикладу це інформація про клієнтів, яким раніше видавалися кредити на різні суми, та інформація про їх погашення.

На підставі навчальної вибірки будується модель визначення значення залежної змінної. Її також називають функцією класифікації або регресії. Для отримання максимально точної функції для навчальної

вибірки висуваються такі основні вимоги:

- Кількість об'єктів, що входять до вибірки, має бути досить велика, адже чим більше об'єктів, тим точніше буде побудована на її основні функція класифікації.
- До вибірки повинні входити об'єкти, що представляють всі можливі класи.
- Для кожного класу в задачі класифікації вибірка повинна містити достатню кількість об'єктів.

На другому етапі побудовану модель застосовують до об'єктів, що аналізуються, тобто до об'єктів з невизначеним значенням залежної змінної [5].

Розглянемо завдання регресії.

За допомогою регресійного аналізу можна отримати конкретні відомості про те, яку форму і характер має залежність між змінними, що досліджуються.

Метод найменших квадратів, що становить математичну основу регресійного аналізу, спочатку застосовувався в астрономії і геодезії. Надалі поєднання методу найменших квадратів та статистичних методів привело до виникнення регресійного аналізу.

Розв’язання завдання регресійного аналізу включає три етапи:

- встановлення форми залежності;
- визначення функції регресії;
- оцінку невідомих значень залежної змінної.

Для будь-яких завдань з кількісними змінними, що змінюються, представляє інтерес дослідження впливу одних змінних на інші. Таким впливом, звичайно, може бути простий функціональна зв'язок між змінними. Але для багатьох фізичних процесів це скоріше виключення, ніж правило.

Ймовірно, часто існує функціональний зв'язок, що занадто складний

для розуміння або для опису простими термінами. У такому разі можна намагатися підібрати апроксимацію цього функціонального зв'язку за допомогою якої-небудь простої математичної функції (прикладом може бути поліном), яка включає відповідні змінні, і згладжувати або апроксимувати «істинну» функцію в певній обмеженій області змін цих змінних.

При дослідженні такої спрощеної функції ми зможемо більше дізнатися про «істинну» залежність, що розглядається, і оцінити окремі або спільні ефекти зміни деяких важливих змінних.

Оцінка значень залежної змінної зводиться до вирішення задачі одного з наступних типів:

- оцінка значень залежної змінної всередині розглянутого інтервалу вихідних даних, при цьому вирішується завдання інтерполяції;
- оцінка майбутніх значень залежної змінної, знаходження значень поза заданого інтервалу вихідних даних, при цьому вирішується завдання екстраполяції.

Обидва завдання вирішуються шляхом підстановки в рівняння регресії знайдених оцінок параметрів значень незалежних змінних. Результат вирішення рівняння представляє собою оцінку значення цільової (залежної) змінної [5].

Кластеризація представляє собою логічне продовженням ідеї класифікації. Це завдання більш складне, адже особливість кластеризації полягає в тому, що класи об'єктів спочатку не визначені. Результатом кластеризації є розбиття об'єктів на групи.

Характеристиками кластера можна назвати дві ознаки:

- внутрішня однорідність,
- зовнішня ізольованість.

Кластеризація відрізняється від класифікації тим, що для проведення аналізу не потрібно мати окрему залежну змінну. Це завдання вирішується

на початкових етапах дослідження, коли про дані мало що відомо. Її вирішення допомагає краще зрозуміти дані, і з цієї точки зору завдання кластеризації – описове завдання.

Для завдання кластеризації характерна відсутність будь-яких відмінностей як між змінними, так і між об'єктами. Навпаки, шукаються групи найбільш близьких, схожих об'єктів. Методи автоматичного розбиття на кластери рідко використовуються самі по собі, в більшості випадків тільки для отримання груп схожих об'єктів. Після визначення кластерів використовуються інші методи інтелектуального аналізу даних, щоб спробувати встановити, що означає таке розбиття, чим воно викликане.

У маркетингових дослідженнях кластерний аналіз застосовується достатньо широко. Це стосується як теоретичних дослідженнь, так і вирішення практичних завдань, наприклад, завдань, що стосуються проблеми згрупування різних об'єктів. При цьому вирішуються питання про групи клієнтів, продуктів і т.д.

Одним з найважливіших завдань при застосуванні кластерного аналізу в маркетингових дослідженнях є аналіз поведінки споживача, а саме: групування споживачів в однорідні класи для отримання максимально повного уявлення про поведінку клієнта з кожної групи і про фактори, що впливають на його поведінку . Об'єднання споживачів в групи дозволяє спростити задачу, так як розглядати поведінку кожного споживача окремо фізично не можливо.

Важливим завданням, яке може вирішити кластерний аналіз, є позиціонування, тобто визначення ланки, в якій потрібно позиціонувати новий продукт, що пропонує ринок. В результаті застосування кластерного аналізу будується карта, за допомогою якої можна визначити рівень конкуренції в різних сегментах ринку і відповідні характеристики товару для можливості попадання в цей сегмент.

Відзначимо ряд особливостей, що притаманні завданню кластеризації.

По-перше, вирішення достатньо сильно залежить від природи об'єктів, що аналізуються. З одного боку, це можуть бути однозначно визначені кількісно об'єкти, а з іншого - об'єкти, які мають нечіткий опис.

По-друге, вирішення завдання великою мірою залежить і від ймовірних відносин об'єктів і кластерів. Також необхідно враховувати такі властивості, як можливість або неможливість приналежності об'єктів до кількох кластерів. Необхідно точне визначення самого поняття приналежності об'єкта кластеру:

- однозначна (належить або не належить);
- ймовірнісна (ймовірність приналежності);
- нечітка (ступінь приналежності) [5].

Розглянемо завдання пошуку асоціативних правил.

В результаті вирішення завдання пошуку асоціативних правил визначаються закономірності між пов'язаними подіями в наборі даних. Знайдені залежності представляють у вигляді правил і можуть бути використані як для кращого розуміння природи даних, що аналізуються, так і для передбачення появи подій.

Спочатку завдання вирішувалося при аналізі тенденцій в поведінці покупців в супермаркетах. Аналізу піддавалися дані про покупки, що робили споживачі. При аналізі цих даних інтерес перш за все представляє інформація про те, які товари споживачі купують разом, які категорії споживачів яким товарам надають перевагу, в який період часу і т.д. Така інформація дозволяє більш ефективно планувати закупівлю товарів, проведення рекламної кампанії і т.д.

У медицині аналізу можуть піддаватися симптоми і хвороби, що спостерігаються у пацієнтів. У цьому випадку знання про те, які поєднання хвороб і симптомів зустрічаються найчастіше, допомагають в майбутньому

правильно поставити діагноз.

Відмінність асоціації від двох попередніх задач інтелектуального аналізу даних полягає в тому, що пошук взаємозв'язків здійснюється між кількома подіями, які відбуваються одночасно [5].

Розглянемо завдання пошуку послідовності.

Різновидом пошуку асоціативних правил є пошук послідовностей, або послідовна асоціація.

Послідовна асоціація дозволяє знайти тимчасові закономірності між подіями. Завдання пошуку послідовності схоже з завданням асоціації, але її метою є встановлення закономірностей не між подіями, що настають одночасно, а між подіями, що впорядковані в часі (тобто відбуваються в деякому порядку). Іншими словами, послідовність визначається як існування високої ймовірності ланцюжка подій, що пов'язані у часі. Фактично, асоціація є окремим випадком послідовності з кроком часу, що дорівнює нулю. Це завдання інтелектуального аналізу даних також називають завданням знаходження послідовних шаблонів.

Правило послідовності може формулюватися так: після події X через певний час відбудеться подія Y.

Аналіз послідовності широко використовується, наприклад, в телекомунікаційних компаніях для аналізу даних про аварії на різних вузлах мережі. Інформація про послідовність здійснення аварій може допомогти у виявленні неполадок та попередженні нових аварій. Наприклад, якщо відома послідовність збоїв: {e5, e2, e7, e13, e6, e1, ...}, де e1 - код збою, то на підставі факту появи збою e2 можна зробити висновок про швидку появу збою e7. Знаючи це, можна провести профілактичні заходи, що допоможуть усунути причини виникнення збою. Якщо додатково володіти і знаннями про час між збоями, то можна передбачити не лише факт його появи, а й час, що також не менш важливо.

Пошук послідовності застосовується і в маркетингу. Наприклад,

може бути встановлено, що після покупки квартири мешканці в 60% випадків протягом двох тижнів купують холодильник, а протягом двох місяців в 50% випадків купується телевізор. Вирішення задач, що подібні до даної, широко застосовується в менеджменті, наприклад, при управлінні циклом роботи з клієнтом.

Завдання прогнозування вирішуються в різноманітних областях людської діяльності, таких як наука, медицина, економіка, виробництво та інших.

Прогнозування є важливим елементом організації управління як окремими господарюючими суб'єктами, так і економіки в цілому. Прикладами можуть бути наступні завдання:

- прогнозування руху грошових коштів;
- прогнозування урожайності агрокультури;
- прогнозування фінансової стійкості підприємства.

Типовим в сфері маркетингу є завдання прогнозування ринків. В результаті вирішення даного завдання оцінюються перспективи розвитку кон'юнктури певного ринку, зміни ринкових умов в майбутньому, визначаються тенденції ринку (структурні зміни, потреби покупців, зміни цін). Крім економічної і фінансової сфери, завдання прогнозування ставляться в найрізноманітніших областях: медицині, фармакології, технічних науках.

Розвиток методів прогнозування безпосередньо пов'язаний з розвитком інформаційних технологій, зокрема, із зростанням обсягів даних, що зберігаються, та ускладненням методів і алгоритмів прогнозування, реалізованих в інструментах інтелектуального аналізу даних.

В результаті вирішення завдання прогнозування на основі особливостей історичних даних оцінюються пропущені або ж майбутні значення цільових чисельних показників.

Прогнозування направлено на визначення тенденцій динаміки конкретного об'єкта або події на основі ретроспективних даних, тобто аналізу його стану в минулому та сьогодні. Таким чином, рішення завдання прогнозування вимагає деякої навчальної вибірки даних.

У найзагальніших рисах рішення задачі прогнозування зводиться до вирішення таких підзадач:

- вибір моделі прогнозування;
- аналіз адекватності та точності побудованого прогнозу.

Для вирішення таких завдань широко застосовуються методи математичної статистики, нейронні мережі і т.д [5].

2.3. Методології ведення проектів інтелектуального аналізу даних

Для вирішення описаних вище завдань існують дві популярні методології ведення проектів інтелектуального аналізу даних:

- Knowledge Discovery in Databases (KDD);
- Cross Industry Standard Process for Data Mining (CRISP-DM).

Технологія Knowledge Discovery in Databases включає в себе питання:

- підготовки даних;
- вибору інформативних ознак;
- очищення даних;
- застосування методів інтелектуального аналізу даних;
- післяобробки даних;
- інтерпретації отриманих результатів.

Безумовно, "серцем" всього цього процесу є методи інтелектуального аналізу даних, що дозволяють виявляти знання.

Весь процес KDD представлений у вигляді схеми на рисунку 2.2.

Рисунок 2.2 - Процесс Knowledge Discovery in Databases

Процес Knowledge Discovery in Databases складається з наступних кроків:

- Підготовка вихідного набору даних. Цей етап полягає в створенні набору даних, в тому числі з різних джерел, вибору навчальної вибірки і т.д. Для цього повинні існувати розвинені інструменти доступу до різних джерел даних. Бажано мати підтримку роботи зі сховищами даних і наявність семантичного шару, що дозволяє використовувати для підготовки вихідних даних не технічні терміни, а бізнес-поняття.
- Передобробка даних. Для того, щоб ефективно застосовувати методи інтелектуального аналізу даних, слід звернути увагу на питання передобработки даних. Дані можуть містити пропуски, шуми, аномальні значення і т.д. Крім того, дані можуть бути надлишкові, недостатні і т.д. У деяких задачах потрібно доповнити дані деякою

апріорною інформацією.

Якщо на вхід системи подати дані в існуючому вигляді, то, зрозуміло, що на виході ми одразу не отримаємо корисні знання. Дані повинні бути якісні і коректні з точки зору використовуваного методу інтелектуального аналізу даних. Тому перший етап KDD полягає в передобробці даних. Більш того, іноді розмірність початкового простору може бути дуже велика, і тоді бажано застосовувати спеціальні алгоритми зниження розмірності.

- Трансформація, нормалізація даних. Цей крок необхідний для приведення інформації до придатного для подальшого аналізу виду. Для цього потрібно виконати, наприклад, приведення типів, квантування тощо. Крім того, деякі методи аналізу вимагають, щоб вихідні дані були в якомусь певному вигляді. Наприклад, нейронні мережі працюють тільки з числовими даними, причому вони повинні бути нормалізованими.
- Інтелектуальний аналіз даних. На цьому етапі застосовуються різні алгоритми для знаходження знань. Це алгоритми кластеризації, нейронні мережі, дерева рішень, встановлення асоціацій і т.д.
- Постобработка даних. Даний етап полягає в інтерпретації результатів та застосуванні отриманих знань в бізнес додатках.

Knowledge Discovery in Databases передбачає послідовність дій, яку необхідно виконати, щоб з вихідних даних отримати знання. Але він не визначає набір методів обробки або алгоритми, що придатні для аналізу. Перевагою технології є те, що даний підхід універсальний і не залежить від предметної області [6].

Cross Industry Standard Process for Data Mining включає шість основних етапів:

- Розуміння бізнесу. Перша фаза процесу спрямована на визначення цілей проекту і вимог з боку бізнесу. Потім ці знання конвертуються

в постановку задачі інтелектуального аналізу даних і попередній план досягнення цілей проекту.

Кроки:

- Визначити бізнес мету.
- Оцінити ситуацію.
- Визначити цілі аналізу даних.
- Скласти план проекту.

- Розуміння даних. Друга фаза починається зі збору даних і ставить за мету познайомитися з даними якомога ближче. Для цього необхідно виявити проблеми, що пов’язані з якістю даних, якщо такі є, зрозуміти, які дані є в наявності, спробувати відшукати цікаві набори даних або сформувати гіпотези про наявність прихованих закономірностей в даних.

  Кроки:

  - Зібрати вихідні дані.
  - Описати дані.
  - Дослідити дані.
  - Перевірити якість даних.

- Підготовка даних. Фаза підготовки даних ставить за мету отримання підсумкового набору даних з вихідних різнорідних та різноформатних даних, що в подальшому будуть використовуватися при моделюванні.

  Завдання підготовки даних можуть виконуватися багато разів без будь-якого наперед заданого порядку. Вони включають в себе відбір таблиць, записів і атрибутів, а також конвертацію і очищення даних для моделювання.

  Кроки:

  - Відібрати дані.
  - Очистити дані.

- Зробити похідні дані.
- Об'єднати дані.
- Привести дані до потрібного формату.

- Моделювання. В даній цій фазі до даних застосовуються різноманітні методики моделювання, будуються моделі, а їх параметри налаштовуються на оптимальні значення. Зазвичай для вирішення будь-якої задачі аналізу даних існує кілька різних підходів. Деякі підходи ставлять особливі вимоги для подання даних. Таким чином досить часто потрібно повернутися на крок назад до фази підготовки даних.

  Кроки:

  - Вибрати методику моделювання.
  - Зробити тести для моделі.
  - Побудувати модель.
  - Оцінити модель.

- Оцінка. На цьому етапі проекту модель вже побудована і отримані кількісні оцінки її якості. Перед впровадженням моделі необхідно переконатися, що всі поставлені бізнес-цілі були досягнуті.

  Основна мета етапу - пошук важливих бізнес-задач, яким не було приділено належної уваги.

  Кроки:

  - Оцінити результати.
  - Зробити перевірку процесу.
  - Визначити наступні кроки.

- Розгортання. Фаза розгортання залежить від вимог та, відповідно до них, може бути простою (наприклад, складання фінального звіту) або складною (наприклад, автоматизація процесу аналізу даних для вирішення бізнес задач). Зазвичай фазу розгортання виконує клієнт. Однак, навіть якщо аналітик не бере участь в розгортанні, важливо,

щоб клієнт чітко розумів, що йому потрібно зробити для того, щоб почати використовувати отриману модель.

Кроки:

- Запланувати розгортання.
- Запланувати підтримку і моніторинг розгорнутого рішення.
- Зробити підсумковий звіт.
- Зробити огляд проекту.

Відповідно до загальних принципів і методологій отримаємо наступний алгоритм виконання інтелектуального аналізу даних:

- Постановка завдання аналізу.
- Збір даних.
- Підготовка даних (фільтрація, доповнення, кодування).
- Підбір параметрів, вибір моделі і алгоритму навчання.
- Навчання моделі.
- Аналіз якості навчання, якщо незадовільний перехід на п. 3 або п. 4.
- Аналіз виявлених закономірностей, якщо незадовільний перехід на п.1, 3 або 4.

Всі етапи життєвого циклу представлені у вигляді схеми на рисунку 2.3.

Можливе переміщення вперед та назад між фазами. Залежно від результату фази або її підзадачі приймається рішення, до якої фази переходити далі. Стрілки показують найбільш важливі і часті переходи між фазами.

Зовнішнє коло символізує циклічну природу аналізу даних. Процес аналізу даних триває і після фази розгортання. Знання, отримані під час процесу, можуть породити нові більш тонкі питання бізнесу.

Подальший процес аналізу даних вигідно проводити, використовуючи знання, що були отримані раніше [7].

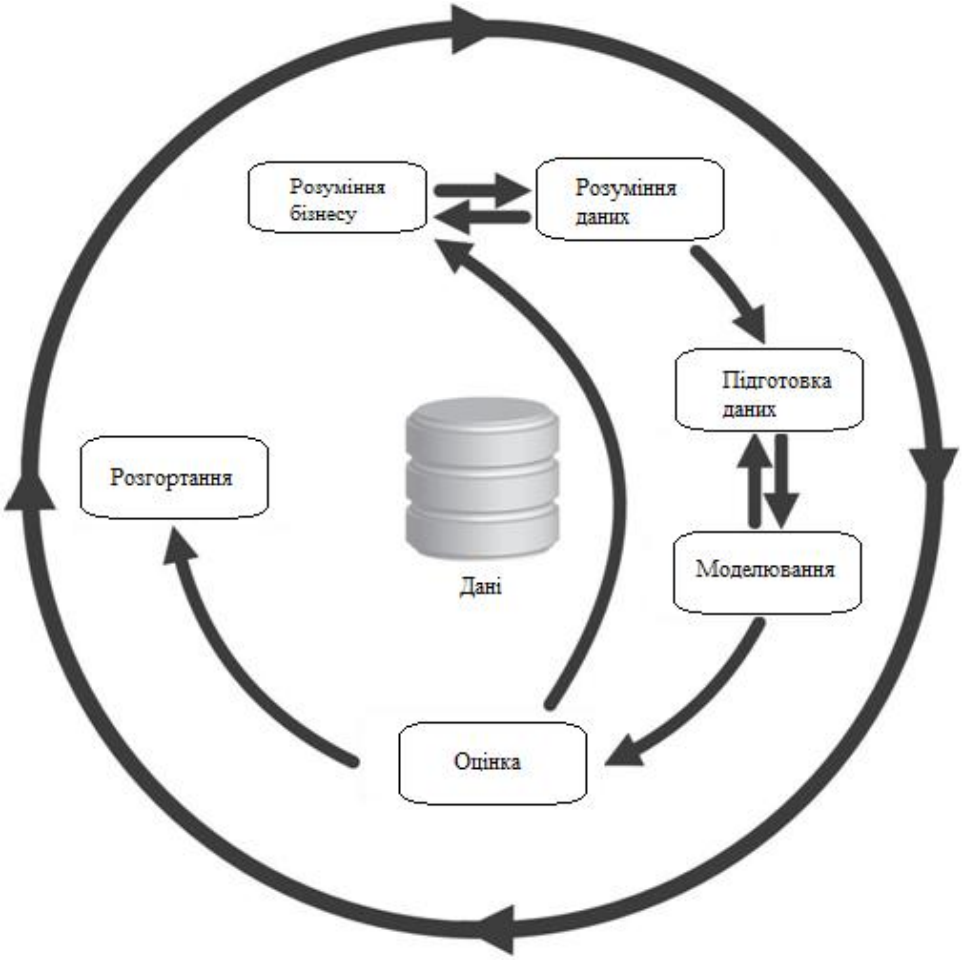


Рисунок 2.3 - Життєвий цикл дослідження даних CRISP-DM

2.4. Процес інтелектуального аналізу даних

Процес інтелектуального аналізу даних є свого роду дослідженням. Як будь-яке дослідження, цей процес складається з певних етапів, що включають елементи порівняння, типізації, класифікації, узагальнення, абстрагування, повторення.

Даний процес нерозривно пов'язаний з процесом прийняття рішень. В процесі інтелектуального аналізу даних будується модель, що потім експлуатується в процесі прийняття рішень.

Розглянемо традиційний процес інтелектуального аналізу даних. Він включає наступні етапи:

- аналіз предметної області;

- постановка задачі;
- підготовка даних;
- побудова моделей;
- перевірка та оцінка моделей;
- вибір моделі;
- застосування моделі;
- корекція та оновлення моделі.

Етап 1. Аналіз предметної області

Дослідження - це процес пізнання певної предметної області, об'єкта або явища з певною метою.

Процес дослідження полягає в спостереженні властивостей об'єктів з метою виявлення важливих, з точки зору суб'єкта-дослідника, закономірних відносин між показниками даних властивостей.

Рішення будь-якої задачі в сфері розробки програмного забезпечення має починатися з вивчення предметної області. Предметна область - це обмежена область реальної дійсності, що підлягає опису або моделювання та подальшому дослідженню. Предметна область складається з об'єктів, що розрізняються за властивостями і знаходяться в визначених відносинах між собою.

У процесі вивчення предметної області повинна бути створена її модель. Знання з різних джерел повинні бути формалізовані за допомогою будь-яких засобів.

Від того, наскільки вірно змодельована предметна область, залежить успіх подальшої розробки програми інтелектуального аналізу даних.

Етап 2. Постановка задачі

Постановка задачі інтелектуального аналізу даних включає наступні кроки:

- формулювання завдання;
- формалізація завдання.

Постановка завдання включає також опис статичної та динамічної поведінки досліджуваних об'єктів.

Етап 3. Підготовка даних

Мета даного етапу - це розробка бази даних для інтелектуального аналізу даних.

Підготовка даних є найважливішим етапом, від якості виконання якого залежить можливість отримання якісних результатів усього процесу інтелектуального аналізу даних. Крім того, слід пам'ятати, що на етап підготовки даних, за деякими оцінками, може бути витрачено до 80% всього часу, відведеного на проект.

Розглянемо кроки даного етапу:

1. Визначення та аналіз вимог до даних.

   На цьому етапі здійснюється так зване моделювання даних, тобто визначення і аналіз вимог до даних, які необхідні для здійснення інтелектуального аналізу даних. При цьому вивчаються питання розподілу користувачів; питання доступу до даних, які необхідні для аналізу, потреби у зовнішніх і внутрішніх джерелах даних; аналітичні характеристики системи.

2. Збір даних.

   Джерелом для вихідних даних є оперативні, довідкові та архівні бази даних, тобто дані з існуючих інформаційних систем. В процесі підготовки даних аналітики і розробники не повинні прив'язуватися до показників, які є в наявності, і описати максимальну кількість факторів і ознак, що впливають на процес, що аналізується. На цьому етапі здійснюється кодування деяких даних.

   При визначенні необхідної кількості даних слід враховувати фактор того, чи є дані впорядкованими або ні. Якщо дані впорядковані і ми маємо справу з тимчасовими рядами, бажано

знати, чи включає такий набір даних сезонну/циклічну компоненту.

Якщо дані не впорядковані, тобто події з набору даних не пов'язані за часом, під час збору даних слід дотримуватися наступних правил.

Недостатня кількість записів в наборі даних може стати причиною побудови некоректною моделі. З точки зору статистики, точність моделі збільшується зі збільшенням кількості досліджуваних даних. Алгоритми, що використовуються для побудови моделей, повинні бути масштабованими.

За умови використання багатьох алгоритмів необхідне визначене співвідношення вхідних змінних та кількості спостережень. Кількість записів (прикладів) в наборі даних має бути значно більшою ніж кількість чинників (змінних).

Набір даних повинен бути репрезентативним і представляти якомога більше можливих ситуацій.

3. Попередня обробка даних.

Аналізувати можна як якісні, так і неякісні дані. Результат буде досягнутий і в тому, і в іншому випадку. Для забезпечення якісного аналізу необхідно проведення попередньої обробки даних. Дані, отримані в результаті збору, повинні відповідати певним критеріям якості.

Етап 4. Побудова моделей

Після закінчення етапу підготовки даних можна переходити до побудови моделі.

Для побудови моделей використовуються різні методи і алгоритми інтелектуального аналізу даних.

Деякі завдання можуть бути вирішені за допомогою моделей, побудованих на основі різних методів. Ідеальної моделі, яка б дозволила

вирішувати різноманітні завдання, не існує. Тому багато розробників включають в інструменти інтелектуального аналізу даних можливість побудови різних моделей. Деякі інструменти інтелектуального аналізу даних створюються спеціально для використання в особливих умовах.

Серед великої різноманітності методів інтелектуального аналізу даних повинен бути обраний метод або ж комбінація методів, при використанні яких побудована модель буде найкращим чином описувати досліджуваний об'єкт.

Іноді для виявлення шуканих закономірностей потрібне використання декількох методів і алгоритмів. В такому випадку одні методи використовуються на початку моделювання, інші - на подальших етапах.

Вибір методу, на основі якого буде побудована модель, повинен здійснюватися з урахуванням постановки задачі, особливостей набору вихідних даних, специфіки завдання, що розв’язується, результатів, які повинні бути отримані на виході.

Етап 5. Перевірка та оцінка моделей

Перевірка моделі має на меті перевірку її достовірності або адекватності. Ця перевірка полягає у визначенні ступеня відповідності моделі реальності.

Адекватність моделі перевіряється шляхом тестування. Адекватність моделі - відповідність моделі модельованому об'єкту або процесу.

Поняття достовірності та адекватності є умовними, оскільки ми не можемо розраховувати на повну відповідність моделі реальному об'єкту, інакше це був би сам об'єкт, а не модель. Тому в процесі моделювання слід враховувати адекватність не моделі взагалі, а саме тих її властивостей, які є істотними з точки зору проведеного дослідження.

Оцінка моделі має на увазі перевірку її правильності. Оцінка побудованої моделі здійснюється шляхом її тестування. Тестування моделі

полягає в «прогоні» побудованої моделі, заповненої даними, з метою визначення її характеристик, а також в перевірці її працездатності.

Побудовані моделі рекомендується тестувати на різних вибірках для визначення їх узагальнюючих здібностей. В ході експериментів можна варіювати обсяг вибірки, набір вхідних і вихідних змінних, використовувати вибірки різної складності.

Етап 6. Вибір моделі

Якщо в результаті моделювання було побудовано кілька різних моделей, то на підставі їх оцінки ми можемо здійснити вибір кращої з них. В ході перевірки та оцінки різних моделей на підставі їх характеристик, а також з урахуванням думки експертів, потрібно вибрати найкращу. Досить часто це виявляється непростим завданням.

Основні характеристики моделі, які визначають її вибір, - це точність моделі та ефективність роботи алгоритму. У деяких програмних продуктах реалізований ряд методів, розроблених для вибору моделі. Багато з них засновані на так званій "конкурентній оцінці моделей", яка полягає в застосуванні різних моделей до одного і того ж набору даних і наступному порівнянні їх характеристик.

Етап 7. Застосування моделі

Після тестування, оцінки та вибору моделі йде етап застосування моделі. На цьому етапі обрана модель застосовується до нових даних з метою вирішення задач, поставлених на початку процесу інтелектуального аналізу даних. Для класифікаційних та прогнозуючих моделей на цьому етапі прогнозується цільовий (вихідний) атрибут.

Етап 8. Корекція і відновлення моделі

Після певного встановленого проміжку часу з моменту початку використання моделі інтелектуального аналізу даних слід проаналізувати отримані результати, визначити, чи дійсно вона "успішна" або ж виникли проблеми в її використанні.

Для того щоб побудована модель виконувала свою функцію, слід працювати над її корекцією. При появі нових даних потрібно робити повторне навчання моделі. Цей процес називають оновленням моделі.

Існує багато причин, що вимагають навчити модель заново, тобто оновити її, щоб відобразити певні зміни.

Основними причинами є наступні:

- змінилися вхідні дані або їх поведінка;
- з'явилися додаткові дані для навчання;
- змінилися вимоги до форми і кількості вихідних даних;
- змінилися цілі бізнесу, які вплинули на критерії прийняття рішень;
- змінилося зовнішнє оточення або середовище (макроекономіка, політична ситуація, науково-технічний прогрес, поява нових конкурентів і товарів і т.д.).

Причини, перераховані вище, можуть знецінити допущення і вихідну інформацію, на яких ґрунтувалася модель при побудові.

Загалом можна сказати, що важливим етапом в процесі інтелектуального аналізу даних є попередня підготовка даних, в тому числі їх очищення. Від якості підготовлених даних будуть залежати результати всього процесу.

В процесі побудови і вибору моделі інтелектуального аналізу даних слід використовувати різні методи і алгоритми, а також їх поєднання. У разі відсутності досвіду використання краще починати з простіших методів. Далі можна поступово ускладнювати моделі, тобто використовувати складніші методи.

Слід пам'ятати, що процес інтелектуального аналізу даних є ітеративним. За умови неможливості отримання результатів, які експерт предметної області вважає прийнятними, необхідно повернутися на один з попередніх етапів процесу [4].

2.5. Інструменти інтелектуального аналізу даних

Для вирішення завдань інтелектуального аналізу даних існує безліч різних інструментів на основі як мов програмування з відкритим кодом, так і закритих комерційних пакетів.

1) Weka. Система дозволяє безпосередньо застосовувати алгоритми до вибірок даних, а також викликати алгоритми з програм на мові Java.

   Поширюється під ліцензією GNU GPL, проект є відкритим, його розвитком займається світове наукове співтовариство. Інструмент написаний мовою програмування високого рівня Java.

   Дані надходять у вигляді матриці, що складається з ознак. Містить функціонал для підключення до баз даних, для цього використовується Java Database Connectivity. Може приймати і виконувати SQL-запити для отримання вибірки з бази даних.

   Weka має користувальницький інтерфейс Explorer, але та ж функціональність доступна через компонентний інтерфейс Knowledge Flow і з командного рядка. Є окремий додаток Experimenter для порівняння передбачливої здатності алгоритмів машинного навчання на заданому наборі завдань.

   Explorer має кілька панелей для роботи з даними:

   - Панель передобработки (Preprocess panel) дозволяє імпортувати дані з бази, CSV файлу і т. д., а також застосовувати до них алгоритми фільтрації, наприклад, переводити кількісні ознаки в дискретні, видаляти об'єкти і ознаки по заданому критерію.
   - Панель класифікації (Classify panel) дозволяє застосовувати алгоритми класифікації і регресії до вибірки даних, оцінювати передбачливу здатність алгоритмів, візуалізувати помилкові передбачення, ROC-криві та безпосередньо сам алгоритм,

якщо це можливо.

- Панель пошуку асоціативних правил (Associate panel) дозволяє виявити всі значущі взаємозв'язки між ознаками.
- Панель кластеризації (Cluster panel) дає доступ до алгоритму k-середніх, EM-алгоритму.
- Панель відбору ознак (Select attributes panel) дає доступ до методів відбору ознак.
- Панель візуалізації (Visualize) будує матрицю графіків розкиду, дозволяє вибирати і збільшувати графіки тощо.

2) Rattle. Забезпечує виконання всіх етапів інтелектуального аналізу даних, що відповідають стандарту CRISP-DM. Використовує мову R - мову програмування для статичної обробки даних.

   Rattle підтримує завантаження джерел даних безлічі типів, однак програма може працювати тільки з одним джерелом, крім того Rattle не дозволяє масштабувати обчислювальні ресурси і вибирати технологію обчислення.

   У інтерфейсі Rattle користувачам надається можливість вибору і налаштування алгоритмів інтелектуального аналізу даних.

   В даному додатку не потрібна авторизація, оскільки він встановлюється локально, і весь процес аналізу даних виконується на комп'ютері користувача, відповідно немає потреби безпечної передачі даних.

   Rattle групує роботу аналітика по проектам, які той може зберігати на свій комп'ютер, а потім відкривати. Однак порівняння декількох моделей не підтримується.

3) Orange. Інтелектуальний аналіз даних можна проводити, використовуючи графічні складові (візуальне програмування), а також із застосуванням скриптів (Python). Інструмент написаний мовою програмування високого рівня Python.

Основні характеристики:

1. Зберігання вибору користувача.
2. Інтелектуальний вибір каналів зв'язку між відметами.
3. Безліч варіантів візуалізації від гістограм до теплових карт.
4. Набір з більш ніж 100 віджетів.

Відповідно до стандарту CRISP-DM інструмент Orange дозволяє виконувати весь цикл інтелектуального аналізу даних. Він передбачає завантаження даних різних типів, аналіз кількох джерел даних, вибір і налаштовування алгоритмів інтелектуального аналізу даних.

4) RapidMiner. Дозволяє вирішити поставлене завдання інтелектуального аналізу даних з використанням спеціального графічного представлення алгоритмів машинного навчання. Представляє собою повноцінний професійний інструмент для аналізу даних. Представляє собою відкрите ПЗ, яке розповсюджується під ліцензією GNU GPL, що дозволяє вбудовувати її в інші додатки, в тому числі, комерційні.

Програма написана мовою Java і пропонує відомі можливості аналізу даних, реалізовані у вигляді заснованого на шаблонах фреймворка. Володіє дуже серйозним набором алгоритмів для обробки та аналізу, включаючи обробку великих масивів даних.

Пакет має досить оригінальну концепцію: робота з набором даних являє собою процес деревоподібного типу, в який можна, як в конструкторі, додавати різні оператори введення, виведення, обробки, візуалізації, аналізу і т.п.

Функціональні можливості:

1. RapidMiner надає більше 400 операторів для всіх найбільш відомих методів машинного навчання, включаючи введення і виведення, попередню обробку даних і візуалізацію.

2. RapidMiner інтегрує в себе оператори Weka.
3. Є вбудована мова сценаріїв, що дозволяє виконувати масивні серії експериментів.
4. Концепція багаторівневого представлення даних забезпечує ефективну та прозору роботу з даними.
5. Графічна підсистема забезпечує багатовимірну візуалізацію даних і моделей.

5) Scikit-learn. Бібліотека представляє собою реалізацію набору інструментів для аналізу даних і машинного навчання. Містить безліч алгоритмів і методів інтелектуального аналізу даних.

Бібліотека базується на SciPy. Набір науковий бібліотек складається з:

1. NumPy - дозволяє використовувати багатовимірні масиви і виконувати операції над ними. Бібліотека написана на C, що підвищує швидкість роботи.
2. SciPy - доповнює мову безліччю наукових бібліотек для аналізу і роботи з даними.
3. Matplot - дозволяє виконувати виведення графіків різної складності та візуалізацію даних.
4. IPython - спеціалізований додаток, що дозволяє використовувати інтерактивну консоль з підтримкою підсвічування синтаксису і автоматичного доповнення тексту.
5. Sympy - додає методи для виконання операції над символами, методи символьних обчислень.
6. Pandas - дозволяє працювати з базами даних, реалізовувати початковий аналіз і фільтрацію отриманих даних.

Розробники бібліотеки були спрямовані на стабільність і надійність, що дозволяє використовувати її в складних комерційних продуктах. Документація бібліотеки також

деталізована і повністю описує не тільки весь API, але також принципи і показує приклади написання вірного коду (best practices).

Бібліотека написана мовою Python, але більшість функціоналу реалізовано мовою C, щоб збільшити швидкість обробки. Також існує можливість використання багатопоточних обчислень. Для роботи з матрицями (масивами), як уже згадувалося, використовується бібліотека NumPy.

Бібліотека позиціонує себе як один з інструментів саме для навчання на даних. Операції над даними та їх завантаження потрібно виконувати за допомогою інших бібліотек, можливо, це пов'язано з уже існуючими інструментами для роботи з даними, що вже зарекомендували себе, наприклад, Pandas і Numpy.

Ряд популярних задач машинного навчання та аналізу даних, які дозволяє вирішувати scikit-learn:

1. Методи кластеризації не розмічених даних, наприклад, метод k- Means.
2. Методи для оцінки якості моделі і підтримка різних алгоритмів поділу на підвибірки (крос-валідація).
3. Методи для зменшення кількості властивостей в даних для подальшої обробки (візуалізація, відбір ознак), наприклад, метод головних компонент.
4. Методи, що базуються на ансамблях алгоритмів машинного навчання.
5. Методи відбору ознак з текстових і графічних даних.
6. Методи визначення ваги ознак і способи їх впровадження.
7. Способи налаштування моделей з пошуком найкращих параметрів.
8. Методи для нелінійного скорочення розмірності даних.

9. Методи навчання з учителем, з них можна виділити лінійні моделі, дискримінантний аналіз, нейронні мережі, метод опорних векторів і дерева рішень [8].

2.6. Вибір інструмента для розробки моделі

Для роботи з даними було вирішено використовувати мову програмування Python і бібліотеку машинного навчання scikit-learn (sklearn).

В Python основний акцент робиться на продуктивності та можливості легко читати код. Досить велике коло програмістів використовує для аналізу даних і статистичних прийомів саме цю мову.

В Python є велика база бібліотек. Python має велику спільноту розробників, але вона дещо неоднорідна, оскільки мова універсальна. Проте саме наука про дані стрімко займає все більш значні позиції в середовищі розробників, що використовують мову Python.

Однією з бібліотек для роботи з даними є sklearn. Бібліотека sklearn надає реалізацію цілого ряду алгоритмів для навчання з учителем і навчання без учителя. Незважаючи на те, що весь інтерфейс бібліотеки представлений на Python, але використання бібліотек, написаних на C, у внутрішній реалізації деяких частин scikit-learn, дозволяє значно підвищити швидкість роботи. Прикладом може бути використання NumPy для роботи з масивами і для операцій з матрицями або використання LAPACK і LibSVM. Для прискорення роботи використовується Cython. Бібліотека Scikit-learn поширюється під ліцензією "Simplified BSD License" і має релізи для безлічі різних операційних систем, включаючи MacOS і Windows. Розробники бібліотеки заохочують цим комерційне та академічне використання sklearn.

Швидкість роботи оцінювалася не тільки теоретична, але і

практична. В теорії всі інструменти представляли собою швидкі інструментах для інтелектуального аналізу даних.

Дослідження показують, що час роботи інструментів Rapid Miner, Weka, мабуть через те що вони написані мовою Java, більше доби, тоді як час роботи інструментів Python і R - не більше години. Так як швидкість інструментів Python і R виявилася приблизно однаковою, то підставами для вибору стали більш звичними, R - функціональна мова програмування, де принципово інший підхід до програмування. В результаті залишилися два інструменти і основною особливістю бібліотеки sklearn є її гнучкість та можливість розширювати функціонал. Саме через свою швидкість роботи і гнучкість були вибрані технології Python і sklearn [9].

3. РОЗРОБКА МОДЕЛІ КЛАСИФІКАЦІЇ ВЕБ-СТОРІНОК

3.1. Класифікація веб-сторінок

Проект, що розробляється повинен відповідати таким вимогам: отримана моделі повинна відносити веб-сторінку до однієї з 15 категорій з точністю передбачення більше 70%.

Для оцінки точності класифікації скористаємося крос-валідацією, де для розрахунку використовується матриця неточностей (таблиця 3.1).

Таблиця 3.1 - Матриця неточностей

		Вірні результати	
		1	0
Результат моделі	1	TP	FP
	0	FN	TN

У таблиці 3.1 міститься інформація про кількість вірно і невірно встановлених значень категорій:

- TP - істинно-позитивний;
- TN - істинно-негативний;
- FP - хибно-позитивний;
- FN - хибно-негативний.

Вимірювання точності класифікації виконується з використанням наступних метрик:

- accuracy;
- precision;
- recall;
- F1 score.

Метрика accuracy показує частку документів, за якими модель класифікації приймає правильне рішення.

$$A(accuracy) = \frac{tp + fp}{N} \quad (1)$$

де tp - істинно-позитивний, fp - хибно-позитивний, N - розмір навчальної вибірки.

Метрика precision характеризує, скільки отриманих від моделі класифікації позитивних відповідей є правильними. Однак це не дає уявлення про те, чи всі правильні відповіді повернула модель класифікації.

$$P(precision) = \frac{tp}{tp + tn} \quad (2)$$

де tp - істинно-позитивний, tn - істинно-негативний.

Метрика повноти recall характеризує здатність моделі класифікації «вгадувати» якомога більше число позитивних відповідей з очікуваних. Можна відзначити, що хибно-позитивні відповіді ніяк не впливають на цю метрику.

$$R(recall) = \frac{tp}{tp + fn} \quad (3)$$

де tp - істинно-позитивний, fp - хибно-позитивний.

Precision і recall надають повну оцінку моделі класифікації. Зазвичай при побудові подібного роду систем доводиться весь час балансувати між цими двома метриками. Якщо підвищити recall, роблячи модель класифікації більш «оптимістичною», це призводить до падіння precision через збільшення числа хибно-позитивних відповідей.

Якщо ж змінювати модель класифікації, роблячи її більш «песимістичною», наприклад, суворіше фільтруючи результати, то зростання precision одночасно призведе до падіння recall через бракування якогось числа правильних відповідей.

$$F_1 = 2 * \frac{P * R}{P + R} \quad (4)$$

Метрика F1 score досягає свого максимуму 1 (100%), якщо P = R = 100%. Величина F1 score є однією з найпоширеніших метрик для

подібного роду систем.

В обчисленні F1 score для завдання класифікації є два основні підходи:

- Сумарний F1 score: результати по всіх класах зводяться в одну єдину таблицю, по якій потім обчислюється метрика F1 score.
- Середній F1 score: для кожного класу формується своя таблиця розподілу і своє значення F1 score, потім береться просте арифметичне середнє для всіх класів.

При класифікації веб-сторінок існує по кілька представників категорій з боку як забороненого, так і дозволеного контенту. В такому випадку метрика precision дозволить більш якісно оцінити модель, так як оцінка виконується за кількістю помилкових влучень, тобто необхідно виконати точну класифікацію для кожної з категорії, а всі інші відмітити як «невідомі» і в подальшому передати на перевірку експертам. При мультикласовій класифікації використовуються метрики усереднення macro-averaging і micro-averaging:

$$P_{micro} = \frac{\sum_{i=1}^{k} tp_i}{\sum_{i=1}^{k} (tp_i + fp_i)}, \quad (5)$$

де tp - істинно-позитивний, fp - хибно-позитивний, k – кількість класів.

$$P_{macro} = \frac{\sum_{i=1}^{k} \frac{tp_i}{tp_i + fp_i}}{k}, \quad (6)$$

де tp - істинно-позитивний, fp - хибно-позитивний, k – кількість класів.

Варіант macro-averaging дає кожному класу однакову вагу в результуючій метриці, а micro-averaging - кожному документу. За умови, коли вага класів однакова, з точки зору вартості помилки, має сенс використовувати macro-averaging. Інакше має сенс використовувати micro-averaging і додати більше документів цього класу в тестову вибірку. У

нашому випадку класи по вартості помилки не рівні, тому для оцінки скористаємося метрикою precision micro-averaging.

При навчанні моделі можлива ситуація, коли виходить досить складна модель, яка точно підходить під дані на навчання, але при цьому дає високу помилку на дані, що не входять в навчання. Уникнути подібної ситуації можна, якщо використовувати крос-валідацію.

Всю вибірку поділяють на дві підвибірки, які можна розбивати за різними принципами: навчальну і контрольну (тестову). Для отриманих підвибірок виконується навчання на алгоритмі машинного навчання, потім оцінюється і обчислюється середня помилка на примірниках контрольної підвибірки. Оцінкою крос-валідації є середня оцінка по всім підвибірках величина помилки на тестових підвибірках.

Якщо вибірка незалежна, то середня помилка крос-валідації дає незміщену оцінку ймовірності помилки. Це помітно відрізняє її від середньої помилки на навчальній підвибірки, яка може виявитися зміщеною (оптимістично заниженою) оцінкою ймовірності помилки, що пов'язано з явищем перенавчання.

До явища перенавчання можна віднести випадок, коли будується дуже складна модель, що дозволяє отримати відмінний результат на навчальній вибірці, але низьку точність при перевірці на тестовій вибірці. Таким чином при використанні крос-валідації можна оцінити рівень перенавчання, а точніше отримати більш правильну оцінку побудованої моделі.

Крос-валідація є стандартною методикою тестування і порівняння алгоритмів класифікації, регресії і прогнозування.

Існує кілька різновидів змінного контролю:

- повна крос-валідація (complete cross validation);
- випадкові розбиття (random cross validation);
- контроль на відкладених даних (hold-out cross validation);

- контроль за окремими об'єктами (leave-one-out cross validation або LOO CV);
- контроль за k блоками (k-fold cross validation);
- контроль за n × k блоками (n × k- fold cross validation).

Керуючись даними «Perception, Sensing & Instrumentation Lab» зазвичай використовується контроль за k блоками або контроль за окремими об'єктами в залежності від кількості вибірки. Відомо, що контроль за k блоками залежить від правильно обраного параметра «k», а при досить великій кількості вибірки краще використовувати контроль за окремими об'єктами [10].

3.2. Збір даних

Одним з найбільш важливих етапів для вирішення завдань за допомогою методів інтелектуального аналізу даних є збір навчальної та тестової вибірки. Так як модель буде модифікуватися, необхідно зберегти дані, щоб вони були доступні для дослідження впливу тих чи інших параметрів. У нашому випадку зберігання в файлі буде не таким зручним, так як необхідно буде використовувати відношення, а в базах даних це дуже просто реалізовано.

При зборі бажано зберігати «сирі» дані, в нашому випадку це вихідний код веб-сторінок. Спочатку було задумано збирати не тільки цільові веб-сторінки, але і їх дочірні сторінки, які можна отримати за посиланнями на веб-сторінки. Виявилося, що це дуже трудомістка за часом задача. Так, якщо одна сторінка завантажується в середньому секунду, то якщо взяти тисячу веб-сторінок з десятьма дочірніми сторінками (посиланнями), ми отримаємо десять тисяч сторінок.

Перед тим, як почати збір даних, необхідно отримати доступ до ресурсу, звідки можна отримати дані. Найчастіше знайти «небажаний» або

заборонений контент можна тільки в тому випадку, якщо безпосередньо займатися подібною діяльністю.

Сайт urlblacklist.com використовується для SquidGuard. SquidGuard - програмний модуль (плагін) для проксі-сервера Squid, призначений для побудови системи фільтрації небажаного веб-контенту. SquidGuard працює за принципом "чорного списку", в якому перераховані небажані сайти, домени та їх ip-адреси. Якщо браузер намагається перейти на таку адресу, то автоматично спрацьовує перенаправлення на сторінку з попередженням або на інший заданий вами сайт. Недолік методу - потрібна підтримка "чорного списку" небажаних сайтів в актуальному стані, тому що постійно з'являються нові і нові сайти. Категорії, які розглядаються в якості заборонених представлені в таблиці 3.2.

Таблиця 3.2 - Заборонені категорії

Категорія	Зміст
alcohol	Інформація про алкогольні продуктах
drugs	Інформація про наркотики
gambling	Інформація про азартні ігри
adults	Порнографічний контент
smoking	Інформація про тютюнову продукцію
violence	Інформація про насилля
weapons	Інформація про зброю
terrorism	Інформація про тероризм
suicide	Інформація про суїцид

Також було отримано список дозволених категорій (таблиця 3.3).

Таблиця 3.3 - Дозволені категорії

Категорія	Зміст
news	Веб-сайти новин
sport	Спортивні веб-сайти
education	Веб-сайти навчальних установ
finance	Економіка і фінанси
shopping	Інтернет-магазини
whitePages	Контент для дітей

Для збору інформації необхідно розібратися в досліджуваній області. Типовий веб-сайт зазвичай складається з набору веб-сторінок, а веб-сторінки зазвичай являють собою текстові файли у форматі *.html. Веб-сторінки можуть містити посилання на інші файли в інших форматах, а також гіперпосилання. Гіперпосиланням називають частину гіпертекстового документа, що посилається на інший елемент в документі, на інший об'єкт або на елементи цього об'єкта.

Для створення HTML-сторінки використовуються HTML-теги. Кожен HTML-тег має своє призначення і дає інформацію браузеру про те, як його слід відображати. Крім HTML-тегів, використовуються ідентифікатори і класи, що дозволяють змінити відображення при використанні каскадних таблиць стилів (CSS). Зазвичай при верстці (написанні HTML коду) структура веб-сторінки розділяється на 3 блоки:

- шапка – header;
- контент – content;
- підвал - footer.

У кожному блоці знаходиться специфічна інформація щодо веб-сайту, така як назва сайту, основний текст і реквізити. Наприклад, зазвичай в якості нижньої частини (підвалу) веб-сторінки встановлюють клас або

ідентифікатор «footer», де може знаходиться назва компанії, авторські права, дублікат основного меню веб-сайту, контактна або коротка інформація про компанію.

Також вводяться теги, що являють собою збагачення семантичного змісту веб-сторінки, такі зміни призводять до уточнення типу мультимедіа об'єктів (відео, зображення, звук і т.д.), розширення універсальних блокових і текстових елементів для визначення інформаційного посилу інформації. Дизайнери веб-сайтів зазвичай усвідомлено використовують стандарт 5-ої версії HTML, тому що такий код простіше читати, а також спрощується його аналіз пошуковими роботами.

Для класифікації веб-сайтів необхідно враховувати безліч факторів, крім тестового контенту. Наприклад, порталам і сайтам новин притаманний матеріал різного змісту, тому на класифікацію впливають дані про батьківські та дочірніх сторінки. Також категорія сайту в цілому може бути визначена тільки за поєднанням єдиного домена. Це необхідно враховувати. Правильна і гнучка вибірка контенту необхідна для роботи, що пов'язана з класифікацією.

Для отримання даних з сайтів необхідно написати модуль. Інформація повинна бути розділена на безліч різних категорій, що дозволить спростити роботу з нею. Орієнтовна класифікація даних:

- посилання (текст в тегу «a» і посилання «href»);
- текст (весь текст сторінки);
- заголовки;
- спеціалізовані теги (meta, head, title);
- теги для конкретизації вибірки (strong, div, теги навігації);
- зображення (шлях «src», текст опису «alt»);
- і т.д.

Завдяки переходу майже всього Інтернету на HTML5 з'явилася можливість розкладання сторінки на семантичні одиниці, що дозволить

спростити класифікацію. Тому в якості додаткової вибірки буде ще кілька категорій:

- контент (вміст «div.content, div # content, article, section»);
- шапка (вміст «#header, .header, header»);
- підвал (вміст «#footer, .footer, footer»);
- навігація (вміст «.nav a, #nav a, #menu a, .menu a, ul a»).

Вибірка виконана з урахуванням використання стандартів як HTML4, так і HTML5 - при наявності необхідного класу або ідентифікатора тег ставиться за стандартом 4-й версії, інакше за стандартом 5-ої версії.

Тепер, коли нам відомі основні можливі групи контенту, можна розглянути варіант збереження отриманих даних. Так як модель буде модифікуватися, необхідно зберегти дані, щоб вони були доступні для дослідження впливу тих чи інших параметрів [2].

У нашому випадку зберігання в файлі буде не таким зручним, так як необхідно буде використовувати відношення, а в базах даних це дуже просто реалізовано. При зборі бажано зберігати «сирі» дані, в нашому випадку це вихідний код веб-сторінок. Так само спрощується пошук, сортування та організація даних. Простим запитом можна отримати необхідну вибірку.

Для зберігання отриманої інформації необхідно розробити структуру бази даних. Таблиці повинні зберігати інформацію про вміст сайту, а також про додаткові дані сторінки, такі як домен, батько і категорії.

Основна інформація буде зберігається в таблиці «pages»:

- url - адреса сторінки;
- html_code - код сторінки;
- html_text - текст сторінки;
- title - заголовок сторінки;
- meta_keyword - ключові слова;

- meta_description – опис;
- p_text - вміст параграфів;
- content - вміст блоку «контент»;
- header - вміст блоку «шапка»;
- footer - вміст блоку «підвал»;
- error - http код відповіді;
- domain_id - ключ на таблицю домену.

Для зберігання в базі даних можна скористатися структурою складається з таблиць:

- domains - таблиця з доменами;
- category_page - таблиця з категоріями веб-сторінок;
- categories - таблиця з категоріями;
- relatives - таблиця зі зв'язками між сторінками;
- pages - таблиця з веб-сторінками;
- tag_ * - таблиці за структурою і змісту ідентичні, де «*»: «a», «h1», «H2», «h3», «decorations», «div», «italic», «img», «nav», «bold», «u» та інші, що містять інформацію з відповідних їм тегами. Схема представлена на рисунку 3.1.

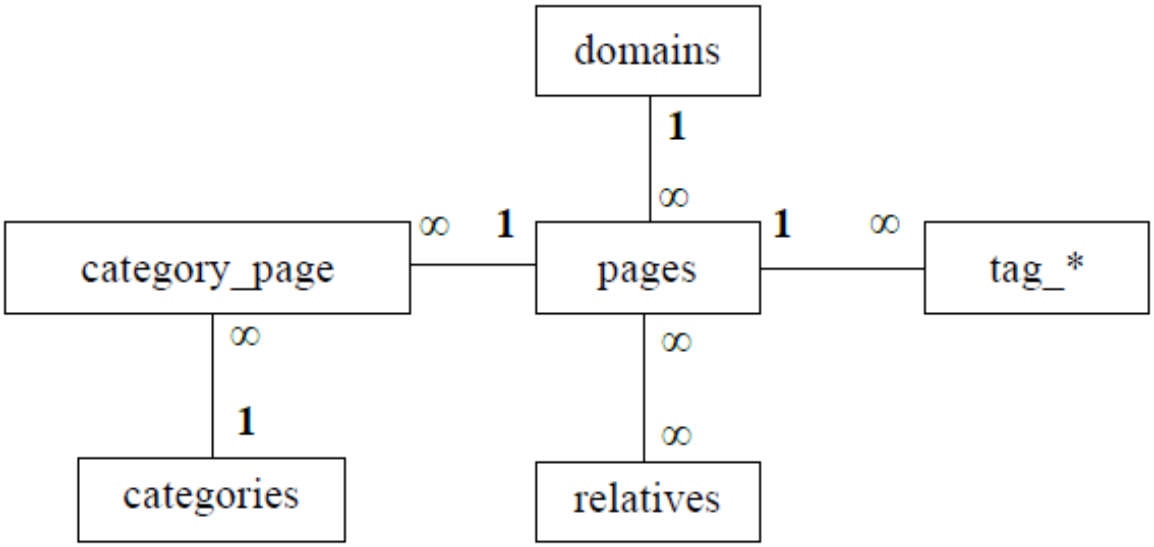


Рисунок 3.1 - Схема структури бази даних

У базі даних ми зберігаємо дані всіх веб-сторінок по групах в окремих таблицях, домени, категорії, що використовуються для класифікації, відношення збережених веб-сторінок, код помилки веб-сторінки.

Для того щоб прискорити процес, можемо скористатися списками адрес веб-сторінок з файлів. У базі зберігаються дані всіх веб-сторінок по групах в окремих таблицях, домени, категорії, що використовуються для класифікації, відношення збережених веб-сторінок, код помилки веб-сторінки.

Для роботи з адресами запропонуємо наступні етапи роботи:

- Зчитування адреси з файлу.
- Збереження адреси в базу даних з кодом помилки «1». Код «1» означає про готовність веб-сторінки до скачування.
- Витягуємо за кодом помилки «1» з бази даних список готових до обробки адрес.
- Кожне посилання передаємо на процес обробки веб-сторінки. Процес обробки веб-сторінки полягає в отриманні, обробці та збереженні даних, що знаходяться за отриманою адресою. Як висновок, процес повертає абсолютні адреси веб-сторінки.
- Отримані абсолютні адреси обробляються тим же алгоритмом, що і батьківські.
- Проводиться перевірка на унікальність адрес нащадків.
- Всі нащадки зберігаються в список на обробку з кодом «[ключ батька] * - 1». Надалі вибираються всі записи з негативним кодом, у яких батьком є значення по модулю цього коду.

В ході роботи була поставлена задача розробки модуля для збору даних. Модуль складається з 4 базових класів:

- AppCrawler;
- Page;

- DbContext;
- ProxyConnection.

Основним класом модуля є AppCrawler, який реалізує інтерфейс IApp і використовує об'єкти класів IPage, DbContext, ProxyConnection.

Запустити модуль можна лише через екземпляр, що успадковує AppCrawler. У конструкторі відбувається ініціалізація об'єкта класу DbContext, і на вхід потрібно передати параметри підключення. Для спрощення роботи з БД використовується клас DbContext. Клас DbContext відповідає за підключення і управління БД, де зберігається інформація про сторінки.

Після створення AppCrawler модуль готовий до роботи і для старту використовується метод launch. Метод launch починає обробку: перевіряє наявність стартового списку посилань, далі передає методу prepareQueue.

Метод prepareQueue відправляє в БД весь поточний список в якості тимчасової черги з кодом «0» і повертає в якості ResultSet для подальшої обробки. Додавання в тимчасову чергу дозволить робити як заплановані зупинки роботи, так і ні.

Після отримання ResultSet відбувається отримання даних про сторінку, передача отриманих даних і нащадків в БД методом parse.

Метод parse працює зі сторінкою і її нащадками. Відправляє нащадків в якості тимчасових сторінок з кодом [ключ батька] * (- 1), що дозволяє вибрати всіх нащадків на наступній ітерації додавання. На вхід отримує сторінку с допомогою методу genRootPage.

Метод genRootPage повертає сторінку, що реалізує інтерфейс IPage. Page - абстрактний клас, що реалізує інтерфейс IPage, відповідає за роботу зі сторінками: отримання даних, обробка адреси, передача даних в БД і обробка HTTP помилок. Даний клас використовує бібліотеку Jsoup для скачування веб-сторінок і вибірки даних з неї.

Існує кілька видів посилань, які не будуть корисні для збору

інформації. Для вирішення подібного завдання використовується метод prepareLink, він фільтрує і приводить до загального вигляду посилання, потім повертає її. Основним «сміттям» в посиланнях є js-скрипти, mailto і хеш-символ. Метод використовується перед роботою з усіма посиланнями, які є тільки отриманими. Він очищає від «сміття» і приводить до вигляду «http(s)://посилання/».

У конструкторі відбувається визначення домену (хоста) сторінки з адреси і додавання в БД. За отримання домену із посилання відповідає метод getDomain.

Метод getDomain дозволяє отримати домен 2-го рівня із посилання. Для цього необхідно вирізати текст між «//» і «/», що містить крапку. При цьому необхідно прибрати параметри GET запиту і перевірити на відповідність домену 1-го рівня. Останнє було реалізовано за допомогою методу «від зворотного» - простіше визначити не домени, наприклад, такі:«asp, aspx, html, php...».

Так як сторінки можуть бути отримані з БД або з Інтернету, то Page залежно від одержуваних параметрів реалізує відповідні способи отримання коду сторінки в конструкторі.

При отриманні сторінки з Інтернету необхідно проводити фільтрацію за типом вмісту: application, audio, video, css, javascript, csv (не включаючи xml), image. В якості успіху отримання даних сторінки - використовується код відповіді HTTP «200». При перенаправленні проводиться зміна посилання по заголовку «Location». У тому випадку, коли відповідь відмінна від 200, відбувається запис в базу даних коду помилки і скасування роботи з поточною сторінкою.

Для отримання і передачі даних в БД з html-коду використовується метод parse. Метод повертає список валідних абсолютних посилань нащадків, робить вибірку даних з поточної веб-сторінки і передає в БД за допомогою класу DbContext. Після передачі даних проводиться фільтрація

посилань підсторінок за допомогою методу getAbsLink.

Метод getAbsLink повертає абсолютне посилання. Посилання можуть бути абсолютні і відносні. Отримати дані за відносними неможливо, тому використовується даний метод для перевірки і при необхідності модифікації відносного в абсолютне посилання. Спочатку проводиться підготовка посилання. Переконавшись, що посилання валідне, намагаємося дізнатися домен. Якщо він присутній, то це абсолютне посилання. У разі якщо домен відсутній, потрібно об'єднати посилання батька і оброблюване, а потім повернути його.

Після додавання вихідного списку посилань та їх даних починається цикл додавання їх нащадків. Для використання проксі сервера в AppCrawler доданий метод setProxy. Метод setProxy встановлює з'єднання з проксі, на вхід приймає екземпляр класу ProxyConnection.

ProxyConnection – клас, що відповідає за підключення і роботу з проксі сервером.

Доступно додавання з різних джерел початкового списку сторінок. Була реалізована можливість зміни виду транзакцій Pages (insert і update записів в таблиці «pages») на основі поліморфізму.

3.3. Навчання моделі

В даному розділі піде мова про методи та способи навчання моделей інтелектуального аналізу даних для класифікації веб-сторінок на основі бібліотеки scikit-learn.

Для роботи з sklearn необхідно встановити безпосередньо інтерпретатор мови програмування python і бібліотеки numpy, scipy. Для роботи з вибіркою бази даних - pandas. Роботу з методами обробки тексту можна здійснити з використанням бібліотеки NLTK. За установку бібліотек відповідає спеціалізований python менеджер - pip. Він дозволяє

не тільки встановлювати необхідні бібліотеки, але також відслідковувати оновлення і залежні бібліотеки під час встановлення.

Працювати з python буде набагато зручніше в середовищі розробки. У нашому випадку буде використана інтерактивна консоль ipython notebook. Інтерфейс середовища представляє собою вікно браузера, в середовищі існує можливість створення в якості проектів Notebook. У проекті є форма для введення коду, але крім цього є можливість додавання декількох форм, кожна з форм є лише одним з етапів скрипта. Дані в даному випадку записуються в оперативну пам'ять, і при запуску наступної форми вона буде використовувати попередні результати обчислень.

Для підключення до бази даних скористаємося встановленим раніше пакетом pandas. Зазвичай робота на мовах програмування з базами даних організована з використанням спеціальних драйверів. В даному випадку використовувалася PostgreSQL база даних, тому необхідно встановити пакет psycopg2.

Після установки драйверів можна приступити до установки з'єднання, де config - це dict з настройками для підключення:

```
conn = psycopg2.connect(host=config["hostname"],
                        user=config["username"],
                        password=config["password"],
                        database=config["database"])
```

Отримане підключення дозволяє працювати з базою, використовуючи курсор. Технічно це дуже незручний спосіб обробки даних в умовах інтелектуального аналізу даних. Тому скористаємося встановленим пакетом pandas, він сформує спеціальний об'єкт типу DataFrame з отриманих даних, над яким можна виконувати різні види операцій, такі як вибірка за умовою, обробка даних, обчислення агрегованих функцій і ін.

3.3.1. Підготовка даних

Так як в нашому випадку в якості даних виступає текст, то для підготовки даних можна розглядати фільтрацію і векторизацію. Фільтрація тексту виконується з метою зменшення шуму, її можна реалізувати двома методами:

- Приведення до загального вигляду - серед різнорідних слів, схожих за коренем або іншими властивостями, слова вирівнюються і спрощуються, тим самим зменшується їх унікальність. Існує ряд позитивних і негативних сторін, які залежать від реалізації методу.

  Можливе формування урізаних слів, які не будуть до обробки мати нічого спільного, але після стануть ідентичні за написанням. З іншого боку, якщо недостатньо сильно фільтрувати, то можна отримати ряд різних за написанням слів, але ідентичних за змістом.
- Видалення слів з низьким смисловим навантаженням - даний метод розглядає використання списку заготовлених слів для видалення, звичайно на видалення відправляються прийменники і артиклі, а також слова зміст яких не впливає на роботу моделі класифікації.

Дані методи можуть використовуватися як окремо, так і спільно. Якість роботи залежить багато в чому від їх реалізації в залежності від завдання і даних.

Для роботи з текстом в бібліотеці sklearn прийнято його оцифрувати і представити у вигляді вектора (векторизувати), для цього є три класи HashingVectorizer, CountVectorizer, TfidfVectorizer.

Основними параметрами при векторизації можна назвати кількість атрибутів (n_features або max_features) і n_gram. Спочатку для можливості оцінки прогресу точності класифікації використовуємо HashingVectorizer з обмеженням атрибутів на 20000 і n_gram (1, 2). Для навчання скористаємося алгоритмом Random Forest. Дані параметри часто рекомендуються на багатьох сайтах, присвячених навчанню

інтелектуального аналізу даних з використанням sklearn для класифікації тексту.

Навчання з вчителем в бібліотеці scikit-learn виконується по єдиному інтерфейсу. Необхідно розділити вибірку, що навчається, на атрибути (ознаки, властивості або features) і мітки (категорії, класи або label), зазвичай в математиці позначаються X, y відповідно. С застосуванням бібліотеки pandas це виконується досить просто:

```
X, y = data.text, data.categories
```

Тепер необхідно розділити на підвибірки для навчання (train) і тестову (test), для цього існує безліч способів. Розподіл вибірки можна виконати наступною конструкцією:

```
X_train, X_test = X [: - 100], X [-100:]
y_train, y_test = y [: - 100], y [-100:]
```

На цьому етапі можна вже виконати навчання моделі з використанням алгоритму Random Forest:

```
from sklearn.ensemble import RandomForestClassifier as rfc
clf = rfc ()
clf.fit (X_train, y_train)
```

Була отримана навчена модель, для перевірки точності скористаємося методом precision:

```
from sklearn.metrics import precision_score pred =
clf.predict (X_test)
scores = precision_score (y_true = y_test,
                          y_pred = pred,
                          average = 'micro')
```

Такий підхід можливий, але скористаємося крос-валідацією k-fold з параметром k = 5. Число вибрано з міркувань оптимальності за часом обробки і одержуваної помилки точності. У sklearn представлено кілька реалізацій методу: KFold, StratifiedKFold, LabelKFold. Найбільше підходить StratifiedKFold, тому що алгоритм являє собою розподіл вибірки

на підвибірки, що містять кожну з категорій. В такому випадку ми отримуємо інший скрипт:

```
from sklearn.cross_validation import cross_val_score,
                                     StratifiedKFold
X, y = data.text, data.category_id
clf = rfc ()
cv = StratifiedKFold (y, n_folds = 5, shuffle = True)
scores = cross_val_score (clf, X, y,
                          cv = cv,
                          scoring = 'precision_micro')
```

В змінній scores зберігаються значення метрики precision_micro для k підвибірок, обчислимо середнє значення точності:

```
avg_score = scores.mean ()
```

Почнемо з фільтрації тексту за наступним алгоритмом:

1. Привести всі символи до одного регістру і видалити «не слова».
2. Виключити загальні слова (стоп-слова).
3. Провести стемінг і лематизацію.
4. Вказати шаблон токенізації (розбиття тексту на слова - токени) і модель n-грами слів (кількість можливих слів в токені).

Для реалізації лематизації і стемінгу скористаємося бібліотекою NLTK. Під лематизацією розглядається процес приведення словоформи до леми - її нормальної (словникової) формі. Стемінг - це процес знаходження основи слова для заданого вихідного слова. Основа слова необов'язково збігається з морфологічним коренем слова.

Крок за кроком виконуємо фільтрацію, а також навчання з крос-валідацією за метрикою precision_micro.

Необхідно привести всі символи до одного регістру методом «lower()». Для видалення всіх «не слів» скористаємося регулярними виразами, в такому випадку шаблон виглядає так: «[^ a-zA-Z]». Точність

моделі з використанням такого фільтра: 0.660.

Лематизація не дозволяє підвищити точність як спільно зі стемінгом, так і окремо. Якщо використовуємо тільки стемінг, то отримаємо модель з точністю: 0.665.

Реалізуємо можливість фільтрації по стоп-словами. Для цього скористаємося вже існуючими списками від Google, MySQL і Word Analytics. Виконаємо об'єднання списків, при цьому прибравши всі перетини і збіги. Для фільтрації по стоп-словам існує функція в бібліотеці sklearn, скористаємося нею і подивимося на отриманий результат. Точність в такому випадку збільшилася до 0.690.

Якщо поглянути на роботу пошуку і видалення стоп-слів з тексту, то можна звернути увагу на те, що багато слів зі списків залишаються в тексті через їх формозміни, наприклад, «sites», де додавання закінчення не змінює кардинально сенсу слова, а значить, його можна було б видалити.

Ще якщо розглянути матрицю векторів, можна звернути увагу на існування спільнокореневих слів, які також об'єднані за змістом. В такому випадку введемо фільтр для очищення закінчень і перевіримо, що з цього вийде. Для фільтрації простіше використовувати також регулярні вирази, але фільтрація буде виконуватися вже окремо від pipline sklearn, шаблон для очищення буде виглядати так: «ing | ly | ed | ious | ies | ive | es | s | ment». В такому випадку збільшуємо точність до 0.698.

Отримана точність моделі недостатня відповідно до технічного завдання. На точність класифікації дуже сильно може вплинути застосований алгоритм машинного навчання, тому для поліпшення можемо замінити алгоритм класифікації і ще раз протестувати модель.

3.3.2. Застосування алгоритмів машинного навчання

Скористаємося раніше описаними алгоритмами машинного навчання на отриманих даних. Всі алгоритми представлені в бібліотеці scikit-learn

для використання.

В результаті:

- kNN - 0.432
- SVM - 0.725
- Logistic Regression - 0.758
- Decision Tree - 0.645

Підбір параметрів в цілому можна назвати емпіричним. Точність моделі класифікації залежить як від векторизації, так і від алгоритму класифікації.

У більшості випадків рішення-переможці на Kaggle представляють собою лінійну комбінацію декількох алгоритмів. Ґрунтуючись на цьому, можна застосувати композицію алгоритмів для проектованої системи.

У машинному навчанні існує кілька алгоритмічних композицій: bagging, boosting, blending, stacked generalization.

Для композиційних алгоритмів використовуються комбінаційні правила: algebraic combiners (mean rule, sum rule і т.д.), voting based methods (majority plurality / voting, weighted majority voting), Borda count, Dempster-Schafer rule і ін.

Використання комбінації алгоритмів на основі blending дозволило досягти кращого результату серед багатьох інших алгоритмів. Основна ідея blending в об'єднанні кількох алгоритмів в один: якщо існують два алгоритму a1 (x) і a2 (x), то їх комбінація представляє собою:

$$a(x) = \alpha a1(x) + (1 - \alpha)a2(x); \alpha \in [0,1]$$

де параметр α вибирається за результатами крос-валідації.

Спробуємо поліпшити модель класифікації, використавши мета-алгоритми (ансамблі). Кращий результат показав алгоритм Bagging з Random Forest: 0.728.

Так як у нас завдання мультикласової класифікації, скористаємося стратегією «one versus rest». Принцип такої стратегії полягає в навчанні

окремих моделей класифікації для кожного з класів і отриманні максимального з них по ймовірності відношення до категорії. Така стратегія також реалізована в бібліотеці sklearn. В результаті точність моделі: 0.763.

На даному етапі ми з'ясували, що застосування Random Forest алгоритму спільно з алгоритмом Bagging і стратегією «one versus rest» дозволяє отримати найкращий результат. Була отримана модель класифікації з точністю, яка підходить встановленим вимогам [11].

3.4. Розробка методів підвищення точності класифікації

Збільшимо точність класифікації веб-сторінок, використовуючи існуючі методики і розробивши власні. Для цього проведемо дослідницьку роботу, присвячену класифікації веб-сторінок на основі методів і алгоритмів інтелектуального аналізу даних.

Одним з найважливіших етапів в підготовці даних на навчання є векторизація. Основними параметрами в цьому випадку виступають алгоритм отримання атрибутів, кількість слів (n-gram) і максимальна кількість атрибутів. У бібліотеці sklearn присутні кілька реалізацій векторизації даних:

- HashingVectorizer (HV) - перетворює вхідний текст в вектор зі значеннями кількості входження слова в текст.
- TfidfVectorizer (TF) - перетворює в вектор зі значеннями відношення числа входження слова до загальної кількості слів документа.

В якості додаткового параметра в TF-IDF можна використовувати сублінійну функцію (SB, sublinear), яка замінює стандартний підрахунок TF і дозволяє прибрати «звичайні» слова з розрахунків. Результати представлені на плакаті №5.

Проаналізувавши отримані результати, можна зробити висновок, що

найвищу точність дає використання векторизації з TF-IDF+ sublinear з 5000 атрибутів і n-gram (1, 2).

При скачуванні даних з мережі Інтернет існує проблема в тому, що у нас досить багато даних і визначити, чи вірно були промарковані веб-сторінки, чи не скінчилася оренда домену і т.д., досить складно. Тому було вирішено провести фільтрацію веб-сторінок за «правильно класифікованими». Для цього векторизуємо всю вибірку, а потім навчаємося на отриманому векторі і робимо прогноз по ньому ж. У цього методу є шанс того, що буде перенавчання, але наявність невірно передбачених менше 5%. Тому приймемо їх за недоступні веб-сайти або категорії, які були спочатку з помилкою і відкинемо їх з вибірки на навчання.

Також стверджується можливість використання PCA (метод головних компонент). Це дозволяє виконати зменшення аналізованої безлічі даних до розміру, який буде оптимальним з точки зору розв'язуваної задачі. Даний метод використовується в якості підготовки даних перед класифікацією.

Найважливішим фактором в інтелектуальному аналізі даних є правильне використання атрибутів. На веб-сторінках можна як атрибути вибрати теги, наприклад, «title», «a», «meta» і блоки «header», «content» і «footer», які були отримані селекторами по тегу, id і class. Так як на цей текст був акцент з боку розробників веб-сайту, то можливо роль їх набагато вища.

Використання окремих моделей класифікації для кожного з атрибутів не дає підвищення точності. Спробуємо поєднати їх з існуючим, тим самим збільшивши кількість входжень ключових слів. В результаті досвідченим шляхом обчислено, що теги «title» і «meta» (description, keywords) збільшують точність.

Для підвищення класифікації спробуємо скористатися іншими

атрибутами. У декількох роботах була вказана можливість використання URL веб-сторінки для її класифікації. В мережі Інтернет для веб-сайтів використовується спрощений формат запису URL:

<схема>: // <хост>: <порт> / <шлях>? <параметри> # <якір>.

Кожна веб-сторінка використовує свій унікальний URL (адресу), при цьому з'явилося неформальне правило «красивого тону» у розробників робити зрозумілі для людини адреси, що містять зрозумілі слова, а не абревіатури або незрозумілі ідентифікатори. Тому використання URL є не тільки швидким способом класифікації. В адресі веб-сторінки можна зустріти основні ключові слова. Скористаємося такою можливістю і відфільтруємо адресу від спеціальних символів і цифр, а також слів, які присутні майже в усіх адресах, такі як розширення, домени першого рівня і протоколи.

Зазвичай адреса складається з декількох слів без поділу, тому скористаємося символьним n-gram. Так як адреси можуть складатися з абревіатур, скорочень або вигаданих слів, скористаємося ще однією властивістю веб-сторінок - заголовком, в ньому зберігається в текстовій формі основна ідея вмісту. Об'єднаймо атрибути, тим самим додамо ключові слова окремих атрибутів, яких не вистачало.

Одним із способів отримання нових атрибутів є метод з використанням word2vec. Дана бібліотека представляє слова у вигляді числового вектора, де мінімальна відстань між векторами буде у найбільш схожих за змістом слів. Для класифікації тексту можна використати в якості атрибута середнє арифметичне (average vector) або набір центроїдів (bag of centroids). В даному випадку точність залежить від розміру вибірки навчання. Можливо, за умови використання більших баз даних, результати можуть бути кращими.

Ще одна складність класифікації полягає у визначенні приналежності тексту, який може належати одночасно до безлічі категорій,

але при цьому ні до однієї з наявних, наприклад, «новини».

Спочатку додамо категорію новини. Виконаємо пошук слів високою вагою і використовуємо в якості ключових слів: «news, finance, sport, political, politics, health, tech, technology, culture, art, weather, ecomony, business, lifestyle, world, national, travel, celebrity, movies , music, fashion».

Використання ключових слів не нове і майже нічим не відрізняється від того, як визначаються інші категорії. Тому введемо критерій оцінки і підрахунку - робити обчислення при наявності хоча б двох входжень основного ключового слова, в даному випадку «news». Таким чином у нас є два рівня ключових слів, де на першому рівні знаходиться основне, а на наступному - інші, за якими виконується вже перевірка. Для збільшення відсотка відповідності використовуємо багатовимірний вектор стоп-слів з використанням синонімів - це дозволяє отримати більш коректний відсоток входження слів без підрахунку слів з однаковим змістом [2].

3.4.1. Зниження рівня помилки в разі невизначеності

При класифікації дуже часто використовується матриця вартості помилки, яка дозволяє провести оцінку результатів класифікації для кожної категорії. При мультикласовій класифікації ще однією проблемою є вибір категорії за відсутності вірної, тому навчання було виконано з використанням присвоювання «невідомої».

Ще одним способом зниження відсотка помилки є додавання в алгоритм передбачення перевірку за обсягом. Якщо розмір досить малий, то категорія веб-сторінки для нас буде «невідомою». Такими веб-сторінками можуть бути заглушки або веб-сторінки, що не містять текст.

Також для підвищення точності додамо використання списку доменів з відомими категоріями.

Подібні рішення перевірити складно на існуючій вибірці через можливе перенавчання - список доменів береться з навчальної вибірки, а

також перевірка за обсягом можлива лише при наявності сторінок в вибірці в якості окремої категорії.

3.4.2. Метод ієрархічної класифікації

Стратегія «one versus rest» дозволила поліпшити точність моделі. Тому використовуємо схожий метод з різницею в тому, що ми можемо використовувати моделі з окремо підібраними параметрами і алгоритмами. Для цього використаємо поодинокі бінарні моделі класифікації, що навчені тільки під свої категорії. Для отримання підсумкової категорії використовуємо метод голосування. Підвищити точність також вдалося за рахунок округлення вихідних даних моделей класифікації.

Метод голосування є не найкращим, тому спробуємо його поліпшити. В якості заміни скористаємося ще однією моделлю класифікації, яка буде навчена за результатами попередніх. В такому випадку необхідно розділити навчання на кілька рівнів. Для початку проводиться розподіл даних на перший (L_1) і другий (L_2) рівень. Вибірка ділиться при повному змішуванні порядку (дозволяє більш коректно визначати точність), при цьому на кожному рівні присутнє рівне співвідношення даних з кожної категорії. Першому рівню встановлюються категорії в бінарному вигляді (0, 1) відповідно до приналежності. При навчанні L_1 (атомарних моделей класифікації) навчаються по кожній категорії з L_1. При навчанні L_2 (рефері) використовуються результати атомарних моделей класифікації (відсоток збігу з категорією). У підсумку на першому рівні ми отримуємо прогноз по кожній категорії, а потім на другому рівні рефері видає остаточне рішення по ним.

Дана методика дозволяє зв'язати кілька моделей класифікації, навчених на різних атрибутах з різними алгоритмами. Використовуючи раніше отримані атрибути, можна побудувати модель для класифікації. Таким чином є можливість вибрати найбільш підходящі алгоритми і

параметри для кожної з категорії і для реферi [12].

3.4.3. Метод класифікації за допомогою «сусідніх» веб-сторінок

Гіпертекстові особливості, такі як посилання, також можуть бути використані в якості атрибутів. Найпростішим способом використання посилання є набір дочірних сторінок в якості атрибутів для навчання, тому спочатку розглянемо можливість їх використання.

Можна спробувати класифікувати цільову веб-сторінку за її дочірніми сторінками, потім методом голосування по отриманому списку результатів класифікації отримати результат. Процес нескладний, але трудомісткий, адже з'являється необхідність завантажувати всі дочірні сторінки цільової веб-сторінки, тому була використана лише частина вибірки.

Для того щоб істотно скоротити часові витрати, можна спробувати обмежити кількість скачуваних дочірніх сторінок. Виконувати збір шести випадкових посилань, де дві з них знаходяться на початку сторінки, в середині та інші в кінці. Для початку проведемо класифікацію по новій вибірці з використанням тільки цільової веб-сторінки. Далі виконаємо класифікацію по дочірнім сторінкам, отримані результати повинні бути нижче через наявність викидів, які обумовлені не тільки змістом різнорідних категорій, а й помилкою при класифікації.

Ще в якості атрибутів можна спробувати використовувати об'єднання тексту всіх дочірніх сторінок. Даний підхід скоріше за все не дозволить досягти бажаного результату через обмеженість дочірніх сторінок, не всі дочірні сторінки можуть бути доступні або відносилися до контенту цільової. Тому розділимо вибірку на підвибірки з усіма дочірніх сторінками.

Проводячи класифікацію тільки за контентом цільової веб-сторінки, крім результатів прогнозів, також можна підраховувати відсоток

упевненості. Це дозволить розділити веб-сторінки за результатами на групи з високим (> ~ 70%) і низьким відсотком впевненості. Розділивши вибірку на ці групи, розглянемо взаємозв'язок з класифікацією по дочірніх сторінках.

Для веб-сторінок з високою впевненістю дочірні сторінки найімовірніше будуть збігатися з цільовою, як для вірно, так невірно класифікованих.

При класифікації веб-сторінок з низькою упевненістю дочірніх сторінок можливість збігу не так очевидна. Проведемо класифікацію по дочірніх сторінках тільки для веб-сторінок з низькою упевненістю, а по цільовим - з високою. Даний метод може дозволити підвищити точність і використовувати новий вид атрибутів, але точність і швидкість отримання результату залежить від кількості дочірніх сторінок.

Зазвичай веб-сторінки з більш складним контентом для класифікації, наприклад, «news» не дають позитивного результату через вміст змішаного набору категорій. Також як і цільові веб-сторінки, що містять зображення, не дозволяють отримати дочірні сторінки. Тому розглянемо можливість використання сусідніх веб-сторінок.

Розглянемо схему того, як можуть бути побудовані зв'язки між веб-сторінками, використовуючи посилання (рис. 3.2).

На схемі розташовані веб-сторінки зі зв'язками на двох рівнях (radius 1-2) в залежності від їх дистанції щодо цільової веб-сторінки (Target Page). На першому рівні цільова веб-сторінка може мати батьківську (Parent) і дочірні (Child) веб-сторінки.

На другому рівні Parent веб-сторінки можуть мати батьківські (Grandparent) і дочірні (Sibling), аналогічно Child веб-сторінки мають дочірні (Grandchild) і батьківські (Spouse).

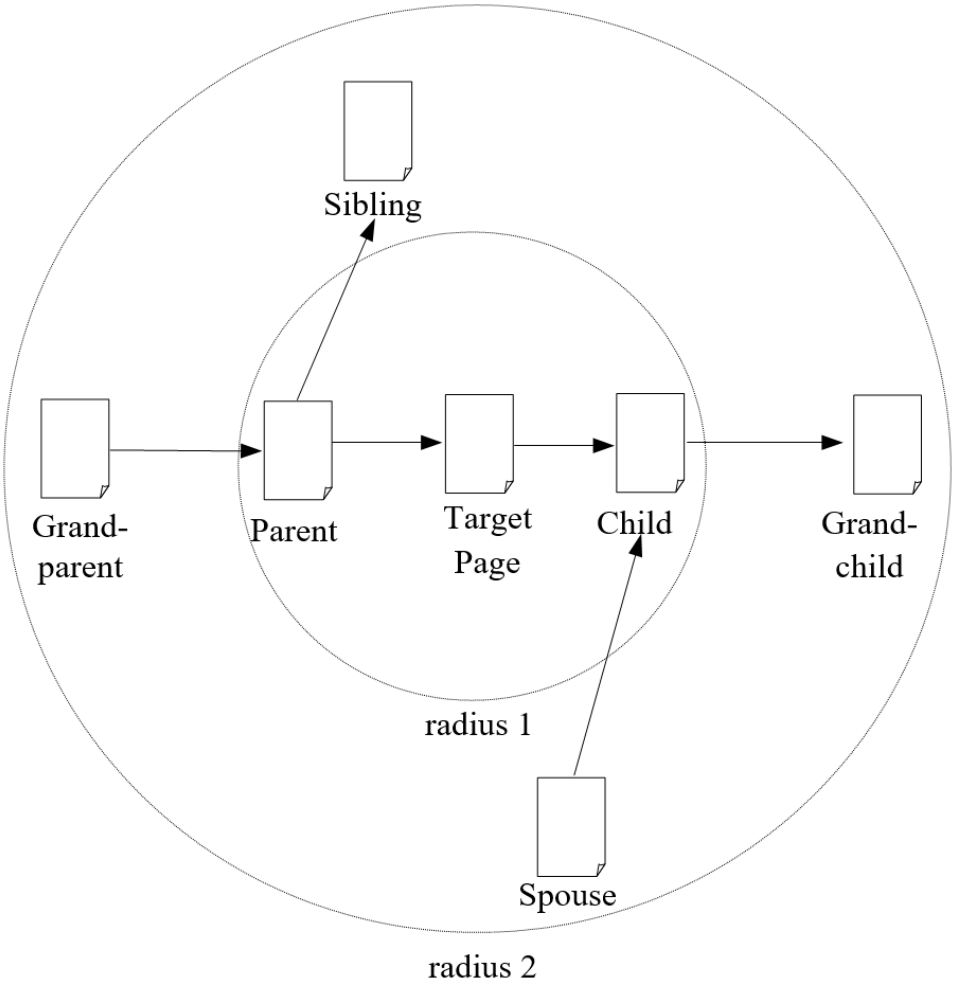


Рисунок 3.2 - Представлення зв'язків між веб-сторінками

На практиці отримати вибірку за подібною схемою можна лише з використанням пошукового робота для збору великої кількості веб-сайтів різних тематик. Такий підхід є практично неможливим за короткий проміжок часу, в той час як інформація, а точніше веб-сторінки з'являються і змінюються щомиті. Практичне застосування даного методу залишається під питанням.

Для отримання батьківських сторінок скористаємося доступними «Backlink» сервісами. Розглядаючи методи скачування, з'явилася ідея підрахунку посилань. Списки відомих доменів веб-сайтів з категоріями можна використовувати в якості атрибутів. Таким чином ми отримуємо вектор, в якому атрибути - колонки з кількістю посилань на категорію [12].

3.5. Оцінка ефективності запропонованих методів

Розглянуті методики були перевірені на всіх доступних алгоритмах класифікації в бібліотеці sklearn.

Методи, засновані на класифікації «сусідніх» веб-сторінок, не дозволяють отримати очікуване поліпшення точності. Можливо, це пов'язано з тим обмеженим набором «сусідніх» веб-сторінок, який використовувався в даному дослідженні. Також процес класифікації, при використанні «сусідніх» веб-сторінок займає чимало часу через необхідність їх скачування. Результати проведеної роботи представлені на плакаті №6.

Якщо розглядати процес класифікації по текстовим даними цільової веб-сторінки, то, як показала практика, одним з найважливіших факторів є використання якісної вибірки, яка не буде містити порожніх або невірно промаркованих веб-сторінок. Це було показано після навчання моделей класифікації на відфільтрованої вибірці по «правильно передбаченим».

Проведені дослідження показують, що найбільш простим і ефективним способом класифікації, що були досліджені в даній роботі, є класифікація на основі ієрархічної моделі з використанням бінарних моделей класифікації з рефері.

ВИСНОВКИ

Мета даної магістерської дисертації - дослідження способів класифікації веб-сторінок за допомогою існуючих методів інтелектуального аналізу даних, модифікація цих методів, підвищення їх точності та розробка моделі, що дозволяє виконувати мультикласову класифікацію веб-сторінок з урахуванням проведених досліджень.

В ході роботи проведено аналіз існуючих методів класифікації веб-сторінок, який дозволив зробити висновок, що такі методи класифікації у повній мірі не задовольняють сучасним вимогам точності та повноти класифікації.

Проведені дослідження показують, що найбільш простим і ефективним способом класифікації, що досліджувалися у даній роботі, є класифікація на основі ієрархічної моделі з використанням бінарних моделей класифікації з рефері. Варто зауважити, що як показала практика, в процесі класифікації одним з найважливіших факторів є використання якісної вибірки, яка не буде містити порожніх або невірно промаркованих веб-сторінок. Такий висновок зроблено після навчання моделей класифікації на відфільтрованої вибірці по «правильно передбаченим».

На основі виконаних досліджень розроблена модель для класифікації веб-сторінок, що дозволяє виконувати мультикласову класифікацію веб-сторінок з оцінкою точності precision з використанням мікроусереднення - 96%.

Також виявлено, що найбільш складно класифікувати веб-сторінки, які не містять тексту. Оскільки зазвичай людина оцінює вміст веб-сторінки на основі зображень, то таких сторінок досить багато. В майбутньому можливе додавання атрибутів такого типу, що допоможе поліпшити якість класифікації. Тому одним з можливих напрямків подальших досліджень може бути використання нових атрибутів, що базуються на зображеннях.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Контент-фильтр - Електрон. дані (1 файл) – Режим доступу: https://ru.wikipedia.org/wiki/Контент-фильтр (дата звернення 10.01.2018) – Назва з екрану.
2. Abdallah T. A., Iglesia B. URL-based web page classification-a new method for URL-based web page classification using n-gram language models // SCITEPRESS Digital Library-KDIR 2014-Intern. conf. on Knowledge Discovery and Information Retrieval. Rome, Italy, 2014.
3. Belmouhcine A., Benkhalifa M. Implicit Links based Web Page Representation for Web Page Classification // Proc. of the 5th Intern. conf. on Web Intelligence, Mining and Semantics. Larnaca, Cyprus, 2015.
4. Чубукова И. А. Data mining - Електрон. дані (1 файл) – Режим доступу: http://lnfm1.sai.msu.ru/~rastor/Books/Chubukova-Data_Mining.pdf (дата звернення 20.02.2018) – Назва з екрану.
5. Солонин Е.Б. Интеллектуальные технологи поиска и анализа данных - Електрон. дані (1 файл) – Режим доступу: http://www.study.urfu.ru/Aid/Publication/13334/1/Solonin.pdf (дата звернення 15.01.2018) – Назва з екрану.
6. Knowledge Discovery in Databases – обнаружение знаний в базах данных - Електрон. дані (1 файл) – Режим доступу: https://basegroup.ru/community/articles/kdd (дата звернення 15.02.2018) – Назва з екрану.
7. CRISP-DM // MachineLearning.ru. Профессиональный информационно-аналитический ресурс - Електрон. дані (1 файл) – Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Crisp-dm (дата звернення 10.02.2018)

8. Завдання Data Mining - Електрон. дані (1 файл) – Режим доступу: http://studopedia.com.ua/1_11366_zavdannya-data-mining.html (дата звернення 20.01.2018) – Назва з екрану.
9. Python - Електрон. дані (1 файл) – Режим доступу: https://ru.wikipedia.org/wiki/Python (дата звернення 15.02.2018) – Назва з екрану.
10. Оцінка класифікатора (точність, повнота, F-міра) - Електрон. дані (1 файл) – Режим доступу: http://bazhenov.me/blog/2012/07/21/classification-performance-evaluation.html (дата звернення 30.01.2018) – Назва з екрану.
11. Sokolova M., Lapalme G. A systematic analysis of performance measures for classification tasks // Information Processing & Management. 2009.
12. Qi X., Davison B.D. Web page classification: Features and algorithms // Journal ACM Computing Surveys. 2009.
13. Data mining - Електрон. дані (1 файл) – Режим доступу: https://ru.wikipedia.org/wiki/Data_mining (дата звернення 10.01.2018) – Назва з екрану.
14. Kwon O.W., Lee J.H. Text categorization based on k-nearest neighbor approach for Web site classification // Inform. Process. Manage. 2003.
15. TF-IDF - Електрон. дані (1 файл) – Режим доступу: https://ru.wikipedia.org/wiki/TF-IDF (дата звернення 10.01.2018) – Назва з екрану.
16. Web mining - Електрон. дані (1 файл) – Режим доступу: https://ru.wikipedia.org/wiki/Web_mining (дата звернення 13.01.2018) – Назва з екрану.
17. Yanchang Zhao. R and Data Mining: Examples and Case Studies. Elsevier, 2012.
18. Інтелектуальний аналіз даних. Класифікація і регресія - Електрон. дані (1 файл) – Режим доступу: http://ukrbukva.net/print:page,1,44734-

intellektual-nyiy-analiz-dannyh-klassifikaciya-i-regressiya.html (дата звернення 30.01.2018) – Назва з екрану.

19. Построение модели и алгоритма кластеризации в интеллектуальном анализе данных - Електрон. дані (1 файл) – Режим доступу: https://cyberleninka.ru/article/v/postroenie-modeli-i-algoritma-klasterizatsii-v-intellektualnom-analize-dannyh (дата звернення 12.02.2018) – Назва з екрану.

Додаток А.

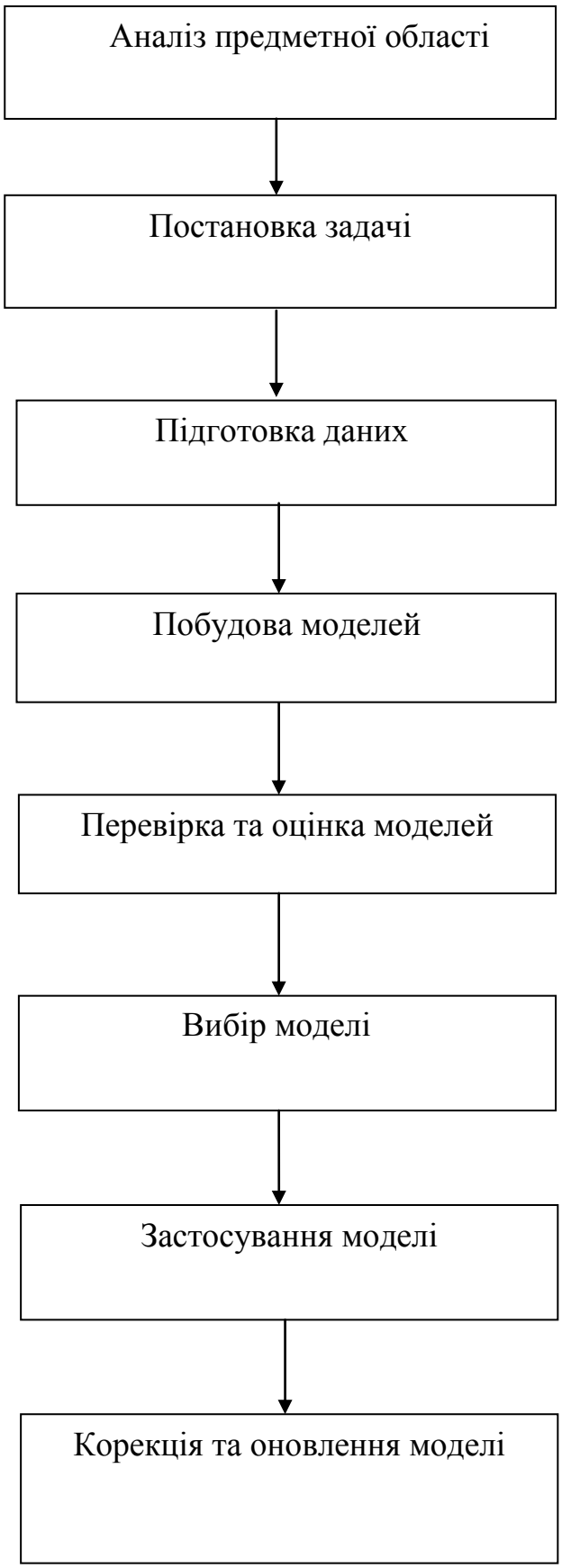
Копії графічного матеріалу

Порівняльна характеристика методів інтелектуального аналізу даних

Алгоритм	Точність	Масштабованість	Інтерпретованість	Здатність до використання	Трудомісткість	Різнобічність	Швидкість	Популярність
Класичні методи	нейтральна	висока	висока/ нейтральна	висока	нейтральна	нейтральна	висока	низька
Нейронні мережі	висока	низька	низька	низька	нейтральна	низька	дуже низька	низька
Методи візуалізації	висока	дуже низька	висока	висока	дуже висока	низька	надзвичайно низька	висока/ нейтральна
Дерева рішень	низька	висока	висока	висока/ нейтральна	висока	висока	висока/ нейтральна	висока/ нейтральна
Поліноміальні нейронні мережі	висока	нейтральна	низька	висока/ нейтральна	нейтральна/низька	нейтральна	низька /нейтральна	нейтральна
k-найближчого сусіда	низька	дуже низька	висока/ нейтральна	нейтральна	нейтральна/низька	низька	висока	низька

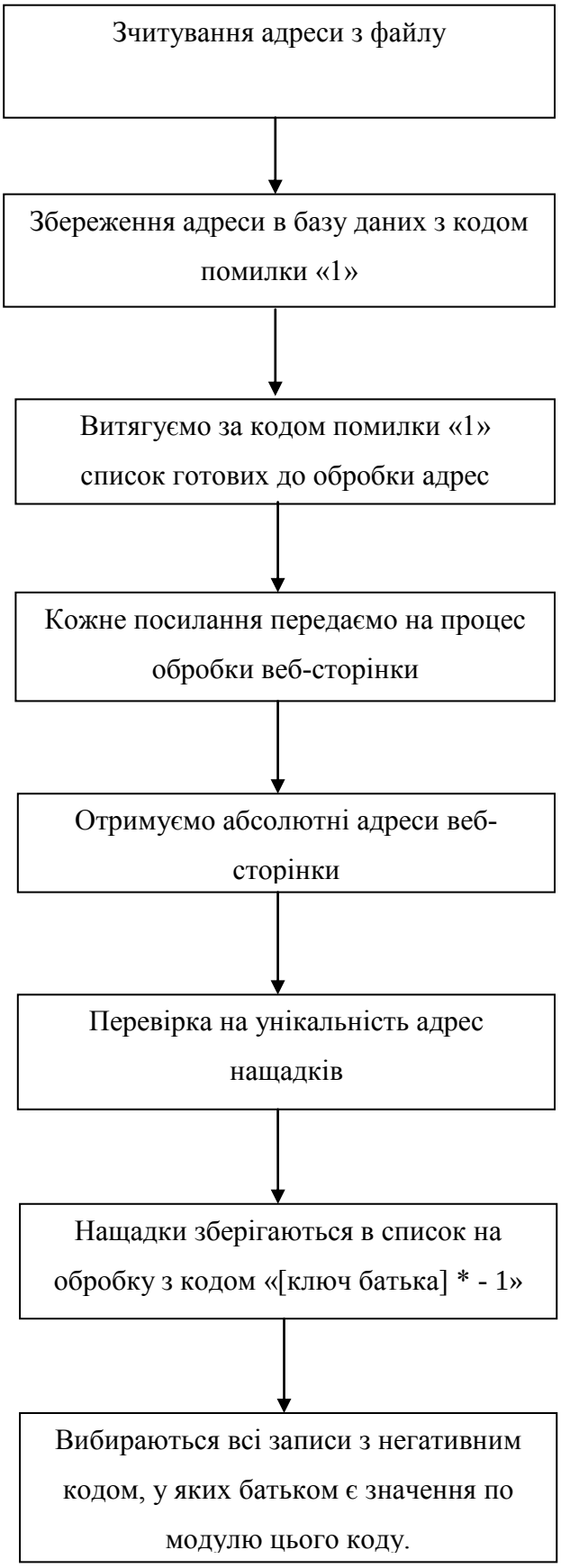
Кузьомка М.А.

Етапи інтелектуального аналізу даних



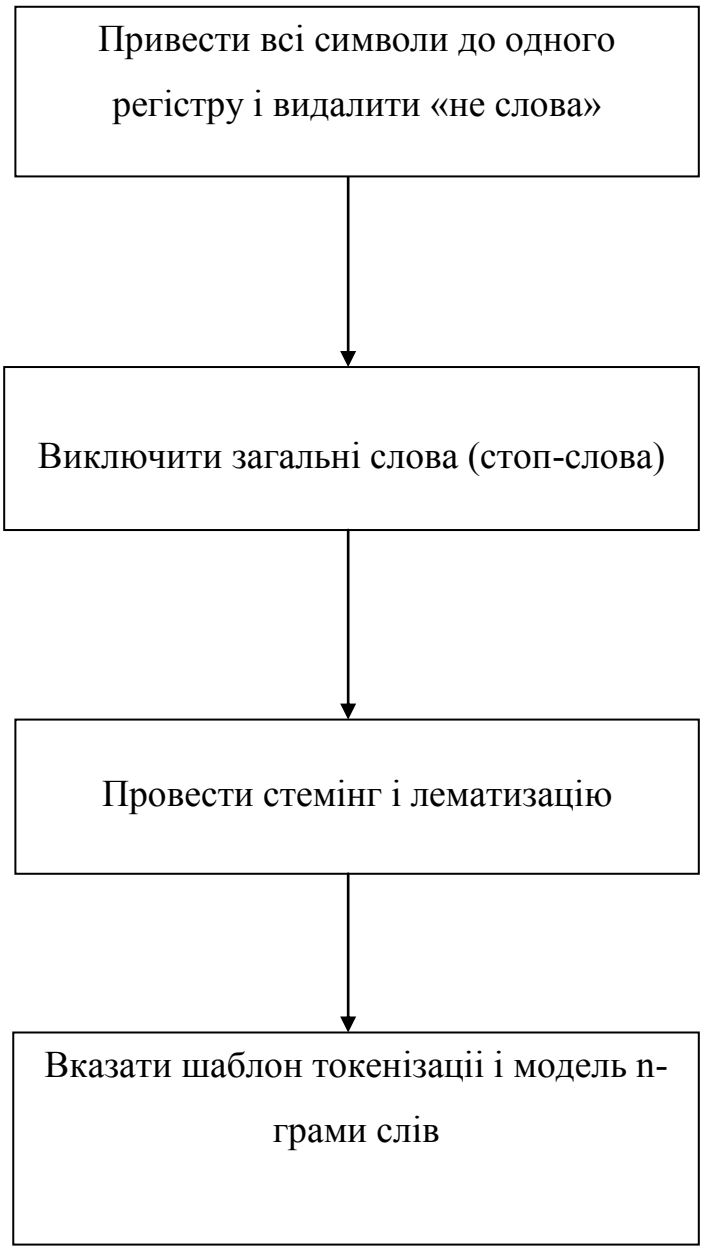
Кузьомка М.А.

Етапи роботи з адресами веб-сайтів



Кузьомка М.А.

Етапи фільтрації тексту



Кузьомка М.А.

Результати векторизації

Число атрибутів	HV	TF	TF+SB
n-gram(1, 2)			
5000	0,747	0,789	0,790
20000	0,767	0,778	0,781
max	0,767	0,772	0,778
n-gram (1, 3)			
5000	0,744	0,788	0,790
20000	0,757	0,783	0,780
50000	0,758	0,772	0,779
n-gram (2, 3)			
5000	0,529	0,687	0,699

Кузьомка М.А.

Порівняльний аналіз методик класифікації

Застосована методика інтелектуального аналізу даних		Точність (precision micro-averaging)
На основі HTML-тегів		0,766
Навчання по «правильно передбаченим»		0,858
Метод головних компонент (PCA)		0,875
На основі заголовка веб-сторінки		0,669
На основі адреси веб-сторінки		0,631
На основі заголовка і адреси веб-сторінки		0,713
Word2vec	Average vector	0,860
	Bag of centroids	0,855
Ієрархічна класифікація	На основі голосування	0,879
	На основі голосування («правильно передбаченим»)	0,928
	Реферi	0,887
	Реферi («правильно передбаченим»)	0,960
Використання «сусідніх» веб-сторінок	На основі голосування за результатами дочірніх сторінок	0,690
	На основі рефері за результатами дочірніх веб-сторінок	0,850
	На основі об'єднаного тексту дочірніх Сторінок	0,770
	З урахуванням «впевненості» дочірніх веб-сторінок	0,851
	На основі зв'язків сусідніх веб-сторінок (Siblings, Spouse)	0,795
	Підрахунок посилань	0,347

Кузьомка М.А

Додаток Б.

Фрагменти програмного коду

classification.py

```
# class of ML classificator based on sklearn
interfaces    from    sklearn.base    import
BaseEstimator, ClassifierMixin from sklearn
import cross_validation
from sklearn import ensemble
from sklearn.externals import joblib
from sklearn.linear_model import LogisticRegression,
LogisticRegressionCV from sklearn.svm import SVC


class MyClassifier(BaseEstimator,
  ClassifierMixin): def_init_(self,
  categories, directory):
    self.atomars = load_atomars(categories, directory)
    self.referee = joblib.load(directory +
    "referee.pkl") self.referee.probability = True # Some
    bug in joblib :( again self.categories = categories

  def
    dig(self,
    x):
    return
    int(x *
    10)

  def fit(self, X, y):
    if self.referie == None:
      self.referee =
      LogisticRegression()
      self.referee.fit(self._L_1_zip_prob
    a(X), y) return self

  def predict(self, X):
    return self.referee.predict(self._L_1_zip_proba(X))
  def predict_proba(self, X):
    return self.referee.predict_proba(self._L_1_zip_proba(X))

  def
    _L_1_proba(se
    lf, X): pred
    = None
    data = []
    for category in
      self.categories: res =
      []
      pred =
      self.atomars[category].predict_proba(X)
      pred_len = len(pred)
      for i in range(pred_len):
        res.append(self.dig(pred[i][1]))
      data.ap
    pend(res)
    return
    data

  def _L_1_zip_proba(self,
    X): return
    zip(*self._L_1_proba(X)
    )
```

```
def load_atomars(categories,
  directory=''): atomars = { }
  for category in categories:
    atomars[category] = joblib.load(directory +
  category+'_.pkl') return atomars
```

classify.py

```
# module of ML

methods
# -*- coding: utf-8 -*-
from nltk import
WordNetLemmatizer import
csv
import re
from flask
importurl_for
import urllib2
from goslate
importGoslate
import textwrap
from os import
path from tld
importget_tld

from translate_yandex import
YandexTranslate from translate_bing
importTranslator

from clf_service import app,

classificator import numpy asnp

from sklearn.feature_extraction.text import
HashingVectorizer from sklearn.feature_extraction.text
import TfidfVectorizer from
sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model importLogisticRegression
from sklearn.externals importjoblib

# if content-length less than _CONTENT_MIN_SIZE we give _P_UNKNOWN_BY_SIZE to
add_confidence
_CONTENT_MIN_SIZE = 10000
_P_UNKNOWN_BY_SIZE = 0.5
# if black_list contains url we give _P_BLACK_LIST to add_confidence
_P_BLACK_LIST = 0.3
_MAX_CONFIDENCE = 89

# list of
categories
CATEGORIES
= [
'alcohol',
'drugs',
'gambling',
'adults',
```

```
'weapons',
'violence',
'smoking',
'suicide',
'terrorism',
'unknown',
'education',
'shopping',
'finance',
'sport']

# @root

APP_ROOT = path.dirname(path.abspath(_file_))


print 'Load clf...'

model = classificator.MyClassifier(list(CATEGORIES), path.join(APP_ROOT,
"ml/model/")) vec =joblib.load(path.join(APP_ROOT, "ml/vec/vec.pkl"))

print "Clf has been loaded"


_ADDITION_CATEGORIES = { "portal" : 0,

"news" : 0 } CATEGORIES += [x for x in

_ADDITION_CATEGORIES.keys()]


# loading
black list def
load_black_lis
t():

  black_list = { }

  for category in
    CATEGORIES:
    pages = []

    with open(path.join(APP_ROOT, "ml/domains/" + category), 'r')
      as f: pages = [line.rstrip('\n') for line in f]

    black_list[category] =
  set(pages) return
  black_list


black_list =

load_black_list()

#region stopwords

loading

def open_sw(filename):

  result = []

  with open(filename, 'rb') as csvfile:

    lines =csv.reader(csvfile,
    delimiter=',') for line inlines:
```

```
            for word in
              line:
              result.app
              end(word)

        return result


    stopwords = open_sw(path.join(APP_ROOT, "ml/stop_words.txt"))
    stopwords_news = open_sw(path.join(APP_ROOT,
    'ml/stop_words_news.txt')) stopwords_portal = ['mail', 'user',
    'sign', 'login', 'logon', 'search'] #endregion



    # adding label with proba
    calculation # proba is
    probability

    # p is threshold of
    probability def
    add_label(p, proba):

      if p > _MAX_CONFIDENCE
        / 100.0: p =
        _MAX_CONFIDENCE /
        100.0
      value = 0
print "p_* : " + str(p)

      for i in
        range(len(proba)
        ): offset =
        proba[i] * p
        value += offset
        proba[i]
        -= offset
        if
        proba[i]
        < 0:
          proba[i] = 0
      proba =

      np.append(proba,

      value) return proba


    #@add_c
    onfiden
    ce #
    title
    is
    title
    # p is threshold of
    probability # proba is
    probabilities
    # return new probabilities
    def add_confidence(title, p,
      proba): index =
      CATEGORIES.index(title)
```

```
  if proba[index] >=
    _MAX_CONFIDENCE: return proba
  value = 0
  # in all proba we take offset 'p'% for category by
  index for i in range(len(proba)):
    if
      ind
      ex
      ==
      i:
      con
      tin
      ue
    offset =
    proba[i] * p
    value +=
    offset
    proba[i] -=
    offset if
    proba[i] < 0:
      proba[i] = 0
  proba[index] =
  proba[index] + value
  return proba

# make labels for UI
# proba is list of
probabilities # return
labels is dictionary
def make_labels(proba):
  cat = CATEGORIES

  prediction = dict(zip(cat, proba))
  s = sorted(prediction.items(), key=lambda x:x[1],
  reverse=True) labels = [(label, round((p*100),2)) for
  (label, p) in s]

  return labels

#check text on keywords with
some vec # vec - is vector
of words
# text - is text
# step - threshold of category
def is_category(text, vec, step=1):
  X_1 = vec.fit_transform([text]).toarray()

  if X_1[0][0]
    <=step:
    return 0
  true = 0
  for x in
    X_1[0][1:
    ]: if x
    >= step:
      true+=1
  proba = round(float(true) / len(X_1[0]) *
  100, 2) return proba
#property get confidence by category
from proba def get_confidence(category,
proba):
```

```
return proba[CATEGORIES.index(category)]

    # predict by page w/ labels
    w/ proxy # as_labels with
    labels
    # proxy_options with proxy
    # return prediction - probabilities
    def predict(page, as_labels=False, proxy_options=None):

      text, translate_success = page_to_text(page,
      proxy_options) text = pre_filter_text(text)

      # additional classificators
      p_news = is_category(text,
      CountVectorizer(vocabulary=stopwords_news), step=2) p_portal =
      is_category(text, CountVectorizer(vocabulary=stopwords_portal))

      p_portal = p_portal +
       p_news/2 if p_portal
        > _MAX_CONFIDENCE:
           p_portal =
         _MAX_CONFIDENCE
      if p_news == 0 and p_portal <
        _MAX_CONFIDENCE: p_portal = 0

      _ADDITION_CATEGORIES["news"] = p_news
      _ADDITION_CATEGORIES["portal"] = p_portal

      print "news " +
      str(p_news) print
      "portal " +
      str(p_portal)

      text = filter_text(text=text, stopwords=set(stopwords),
      is_lem=True) X = vec.transform([text]).toarray()
      predict = model.predict_proba(X)[0]

      #region @add_confidence block
      for category, value in
        _ADDITION_CATEGORIES.iteritems(): print category
        + str(value)
        predict = add_label(value/100.0,

      predict) print predict

      if page.content_length < _CONTENT_MIN_SIZE:
        predict = add_confidence("unknown", _P_UNKNOWN_BY_SIZE, predict)

      #region @black_list
      block domain =
      make_domain(page.url)
      print domain
      for category, domains in
        black_list.iteritems(): if domain in
        domains:
          print domain + "-> in black list -> " + category
          predict = add_confidence(category, _P_BLACK_LIST,
          predict) break;
      #endregion @black_list block

      #endregion @add_confidience

      block
```

```
        if as_labels:
          return make_labels(predict),

        translate_success return predict,

        translate_success

      #get domain
      from link #
      link - string
      URL def
      make_domain(li
      nk):
        try:
if not link.startswith('https://'):

            if not
              link.startswith('http://'
              ): link = "http://" +
              link
          d = get_tld(link, as_object=True,
          fail_silently=True) domain = d.subdomain + "." +
          d.tld
          if
            domain.startswith
            ('.'): domain =
            domain[1:]
          if
            domain.startswith
            ('www'): domain =
            domain[4:]
          return
        domain
        except:
          return None

      # normalize text
      def
        pre_filter_text
        (text): text =
        text.lower()
        letters_only = re.sub("[^a-zA-Z]", " ",
        text) return letters_only

      # remove stopwords and stamming text
      def filter_text(text, stopwords,
        is_lem=False): words = text.split()

        meaningful_w
        ords = []
        for w in
        words:
          w =
          w.st
          rip(
          ) if
          w ==
          "":
            continue
```

```
    w =stem(w,
    is_lem)
    length =
    len(w)
    if w not in stopwords and length >= 3 and length < 25:
      meaningful_words.append(w)

  return "

".join(meaningful_words) #

stemmingmethod
# word - string word
# is_lem -lemmatize
it or not def
stem(word, is_lem):
  stem = word
  regexp =
  r'^(.*?)(ing|ly|ed|ious|ies|ive|es|s|ment)?$'
  stem, suffix = re.findall(regexp, word)[0]
  if is_lem:
    wnl =
    WordNetLemmatizer
    () stem =
    wnl.lemmatize(ste
    m)
  return stem

#
transla
te text
# txt -
text
# lang_to - to lang
# lang_from - from lang
def translate(t, txt, lang_to,
  lang_from): data = []
  result = u""
  parts =textwrap.wrap(txt,
  width=10000) for part inparts:
    result = t._translate(part, lang_to, lang_from)
    data.append(result)
  result = "
  ".join(data)
  return result

# convert
page to text
# page - Page
object
# proxy_options - with proxy or not

def page_to_text(page,
  proxy_options=None): text =u" "
  def_lang_txt =u" "

  text = page.html_text
  def_lang_txt = text[:200] if len(text) > 200 else text
  def_lang_txt += (u" " + page.title)

  err = 0
```

```
  for t_service in
    translate_services: t =
    t_service(proxy_options)
    try:
      lang_from =
      t.detect(def_lang_txt)
      lang_to = "en"
      if lang_to != lang_from:
        text = translate(t, text, lang_to,
      lang_from) break
        except: err += 1 continue
  if err ==
    len(translate_services):
    return text, False
  return text, True

# translate with google
def
  translate_google(proxy_opt
  ions): t = Goslate()
  if proxy_options:
    proxy_handler = urllib2.ProxyHandler(proxy_options)
    proxy_opener = urllib2.build_opener(urllib2.HTTPHandler(proxy_handler),
                  urllib2.HTTPSHandler(proxy_handler))
    t =
  Goslate(opener=proxy_opener)
  else:
    t = Goslate()
  t._translate =
  t.translate print
  "google"
  return t

#translate with bing
def
  translate_bing(proxy_opt
  ions): t =
  Translator("clf_service"
  ,
           "d0yDw3KHIeqemySuGfP60WbtGEpBEfiKrCU/u3aMm4s=",
           proxy_options=proxy_options)
  t.detect =
  t.detect_language
  t._translate =
  t.translate print
  "bing"
  return t

#translate with bing
def
  translate_yandex(proxy_opt
  ions): t =
  YandexTranslate(
'trnsl.1.1.20151011T230549Z.191d91bce8a7a90a.244f4a358ab9b8f392bb60a6239499804df2f828',
 proxy_options=proxy_options
)

  t._translate = lambda text, lang_to, lang_from:
                t.translate(text, lang_from+"-
                "+lang_to)['text'][0]
```

```
        print
        "yandex"
        return t

translate_services = [ translate_yandex, translate_google, translate_bing ]
```

Додаток В.

Публікація за темою роботи

УДК 004.77

К.т.н., ст. науковий співробітник Боярінова Ю. Є., студентка Кузьомка М. А.

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

ДОСЛІДЖЕННЯ КЛАСИФІКАЦІЇ ВЕБ-СТОРІНОК НА ОСНОВІ ТЕХНОЛОГІЇ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ

Abstract

Yulia Boyarinova, PhD; Maryna Kuziomka, student

Research a classification of web-pages based on data mining techniques

This paper examines methods of classification of web-pages that were investigated and analyzed. There was developed a model for classification of web-pages using data mining techniques. For development there were used a combination of existing methods.

Вступ

Сьогодні інтернет займає важливу роль в житті людини. Інформаційний простір в мережі налічує вже мільйони гігабайт даних різного роду і відрізняється високим рівнем доступності для користувачів.

Для автоматизації перевірки і класифікації веб-контенту можна використовувати методи інтелектуального аналізу даних (ІАД). Мета технології ІАД - виявити структури даних і знайти закономірності в слабоструктурованих даних [3].

Інформація в Інтернеті відрізняється високою динамікою: створення нового контенту, його редагування та видалення займають кілька секунд. З огляду на кількість користувачів, які можуть створювати небажаний контент, використання традиційних методів виявлення та класифікації подібної інформації стає незручним.

Постановка задачі

Метою роботи є дослідження способів класифікації веб-сторінок за допомогою існуючих моделей, методів і алгоритмів інтелектуального аналізу даних та підвищення їх точності.

Існуючі рішення

Сьогодні існує безліч робіт в області класифікації веб-сторінок. Основною відмінністю класифікації веб-сторінок від звичайного тексту є гіпертекст. У зв'язку з цим класифікацію можна розділити на класифікацію за вмістом цільової веб-сторінки або за вмістом сусідніх веб-сторінок.

Атрибути веб-сторінок можна розділити на текстові та візуальні. Текстова інформація більш зручна для використання в класифікації. Для цього використовуються кілька варіантів вибору атрибутів, таких, як bag-of-words, TF-IDF і n-gram. Такі методи зазвичай застосовуються в дослідженнях Text Mining. Веб-сторінка використовує HTML-теги як контейнери, в яких може знаходитись текст. Такі теги можуть бути обрані в якості атрибутів. Веб-сторінки можна представити як ієрархією візуальних елементів, таких, як навігація, контент та інші блоки. Але не завжди впровадження такого методу дозволяє збільшити точність. Тому можна поєднати дані підходи для підвищення точності класифікації. Поліпшити продуктивність класифікатора також можливо за рахунок зменшення розміру даних, тому що в такому випадку потрібно аналізувати менший об’єм інформації [2].

Використання сусідніх веб-сторінок дозволяє значно підвищити точність. При аналізі зв’язків між веб-сторінками можна побудувати граф і на основі цього отримати числовий вектор, використовуючи методику як в TF-IDF.

Кожна веб-сторінка має свою унікальну адресу URL, за якою можна виконати швидку класифікацію. Хоча результат в такому випадку набагато нижче, ніж при звичайній класифікації тексту. Використання n-gram при класифікації за URL веб-сторінки може підвищити ефективність [1].

Навчання моделей класифікації

Існує ряд популярних інструментів для обробки отриманих даних: RapidMiner, MATLab, Python, R, Theano, Weka. Можливості кожного приблизно однакові, тому вибір можна вважати суб'єктивним. Було вирішено використовувати Python та бібліотеку sklearn.

При роботі з текстом в бібліотеці sklearn прийнято його оцифрувати і представляти у вигляді вектора, для цього є 3 класи: HashingVectorizer, CountVectorizer, TfidfVectorizer. Основними параметрами при векторизації можна назвати кількість атрибутів і n_gram. Спочатку використовуємо HashVectorizer з обмеженням атрибутів на 20 000 і n_gram (1, 2). Для навчання використовуємо алгоритм Random Forest.

Для вимірювання точності класифікації використаємо крос-валідацію, яка полягає в оцінці аналітичної моделі і її поведінки на незалежних даних та вже реалізована в sklearn. Для розрахунку використовується матриця

похибок. Нехай tp - це true positive, tf - true false, тоді формула для визначення точності (precision) для бінарної класифікації виглядає так:

$$P = \frac{tp}{tp+tf} \quad (1)$$

У разі мультикласової класифікації використовуються значення macro-averaging і micro-averaging:

$$P_{macro} = \frac{\sum_{i=1}^{k} \frac{tp_i}{tp_i + fp_i}}{k}, \quad (2)$$

$$P_{micro} = \frac{\sum_{i=1}^{k} tp_i}{\sum_{i=1}^{k} (tp_i + fp_i)}, \quad (3)$$

де k – кількість класів, fp – false positive.

При обчисленні macro-averaging кожному класу надається однакова вага в результуючій точності, а micro-averaging - кожному документу. Якщо вага класів однакова, з точки зору вартості помилки має сенс використовувати macro-averaging, інакше - micro-averaging і додати більше документів цього класу в тестову вибірку. В даному випадку класи за вартістю помилки не рівні, тому для оцінки користуємось значенням micro-averaging.

Очистимо текст від всіх символів, крім букв, за шаблоном: «[^ a-zA-Z]», точність навченої моделі - 0.660. Ще одним способом фільтрації є стемінг, який полягає в знаходженні основи слова для заданого вихідного слова. Якщо використовуємо стемінг з бібліотеки NLTK, отримаємо модель з точністю 0.664. Реалізуємо можливість фільтрації по стоп-словами - значення за формулою(3) - 0.690. Використовуємо всі фільтри, а потім заберемо закінчення слів «ing | ly | ed | ious | ies | ive | es | s | ment». В такому випадку отримуємо точність 0.698. Спробуємо поліпшити модель додав метакласифікатори та застосував алгоритм Bagging з Random Forest. В цьому випадку точність склала 0.728.

Так як розглядається задача мультикласової класифікації, скористаємося стратегією «one versus rest», головний принцип якої полягає в навчанні окремих моделей категоризації для кожного з класів і отримання максимального з них за ймовірністю відношення до категорії. Така стратегія також реалізована в бібліотеці sklearn. В результаті точність моделі: 0.762.

На даному етапі встановлено, що застосування алгоритму Bagging з Random Forest і стратегією one versus rest дозволяє отримати найкращий результат. Точність отриманої моделі не достатня для застосування на практиці, тому спробуємо поліпшити її, використовуючи існуючі методики класифікації веб-сторінок і запропонував власні.

Використання атрибутів для підвищення точності класифікації

1) Класифікація веб-сторінок на основі тегів

Найважливішим фактором в ІАД є правильне використання атрибутів. На веб-сторінках можна в якості атрибутів вибрати теги «title», «a», «meta» і блоки «header», «content» та «footer», які були отримані за допомогою селекторів тегів, id та class (тобто за формальними описами того елемента або групи елементів, до яких застосовується вказане правило стилю). Використання окремих класифікаторів для кожного з атрибутів не дає підвищення точності. Спробуємо поєднати їх з існуючими, тим самим збільшив кількість вхождень ключових слів. В результаті виявлено, що додавання тегів «title» і «meta» дає точність 0.766.

2) Класифікація веб-сторінок на основі URL

URL є стандартизованим способом запису адреси ресурсу в мережі. В Інтернеті для веб-сайтів використовується спрощений формат запису URL:

<схема>: // <хост>: <порт> / <шлях>? <параметри> #<якір>.

В адресі веб-сторінки можна зустріти основні ключові слова. Скористаємося такою можливістю і відфільтруємо адресу від спеціальних символів і цифр, а також слів, які присутні майже в усіх адресах. Після вибору відповідних параметрів була отримана точність 0.630. Точність досить низька, що очікувано, адже адреси можуть складатися з абревіатур, скорочень або вигаданих слів. Скористаємося ще однією властивістю веб-сторінок - заголовком, в якому зберігається в текстовій формі основна ідея вмісту. Якщо вивести класифікацію за назвою, то точність досягає 0.668.

3) Класифікація веб-сторінок на основі Word2vec

Одним із способів отримання нових атрибутів є метод з використанням Word2vec. Дана бібліотека представляє слова у вигляді числового вектора, де мінімальна відстань між векторами буде у найбільш схожих за змістом слів. Для класифікації тексту можна використовувати в якості атрибута середнє арифметичне або набір центроїдів. Результат класифікації в обох випадках відповідно 0.860 і 0.855.

4) Класифікація веб-сторінок на основі дворівневих ключових слів

Ще одна складність класифікації полягає у визначенні приналежності тексту, який може відноситися одночасно до безлічі категорій, але при цьому ні до однієї з наявних, наприклад «новини».

Спочатку додамо категорію «новини». Виконаємо пошук слів з високою вагою, використовуючи в якості ключових слів news, finance, sport, political, politics, health, tech, technology, culture, art, weather, economy, business, lifestyle, world, national, travel , celebrity, movies, music, fashion.

Використання ключових слів не нове і нічим не відрізняється від того, як визначаються інші категорії. Тому введемо критерій оцінки та

підрахунку - проводимо обчислення за наявності хоча б двох входжень основного ключового слова, в даному випадку «news». Таким чином, є два рівня ключових слів, де на першому рівні знаходиться основне, а на другому - решта слів, за якими виконується перевірка. Для збільшення відсотка відповідності беремо багатовимірний вектор стоп-слів з використанням синонімів - це дозволяє отримати більш коректний відсоток входження слів без підрахунку слів з однаковим змістом [2].

Метод ієрархічної класифікації

Застосування стратегії «one versus rest» дозволило поліпшити якість моделі. Тому використаємо схожий метод, який відрізняється тим, що можна використовувати моделі з окремо підібраними параметрами і алгоритмом. Для цього візьмемо поодинокі бінарні класифікатори, що навчені строго під свої категорії.

Для отримання підсумкової категорії застосуємо метод голосування. Домогтися підвищення точності вдалося за рахунок округлення вихідних даних класифікаторів.Спробуємо покращити метод голосування.

В якості заміни скористаємося ще одним класифікатором, який буде навчений за результатами попередніх. В такому випадку необхідно розділити навчання на кілька рівнів. Для початку дані ділять на перший (L_1) і другий (L_2) рівні. Вибірка ділиться при повному змішуванні порядку, причому на кожному рівні є рівне співвідношення даних з кожної категорії. Першому рівню встановлюються категорії в бінарному вигляді (0, 1) відповідно до приналежності. При навчанні L_1 (атомарних класифікаторів) навчаються по кожній категорії з L_1. При навчанні L_2 (рефері) використовуються результати атомарних класифікаторів. У підсумку на першому рівні отримуємо прогноз по кожній категорії, а потім на другому рівні рефері видає остаточне рішення по ним.

Дана методика дозволяє зв'язати кілька класифікаторів, навчених на різних атрибутах, з різними алгоритмами. Використовуючи раніше отримані атрибути, можна побудувати модель для класифікації. В якості алгоритму підведення підсумку високу точність показав SVM. При класифікації без навчання по «правильно передбаченим» точність 0.887. При класифікації с навчанням по «правильно передбаченим» точність 0.960. Таким чином, вдалося вибрати найбільш підходящі алгоритми і параметри для кожної категорії і для рефері, тим самим підвищивши точність класифікації до 0.960.

Метод класифікації за допомогою «сусідніх» веб-сторінок

Гіпертекстові особливості, такі, як посилання, також можна використовувати в якості атрибутів. Найпростіший спосіб - використання набору підсторінок, як атрибутів для навчання. Можна спробувати класифікувати цільову веб-сторінку за її підсторінками, потім методом голосування за отриманим списком результатів класифікації отримати результат. Процес нескладний, але трудомісткий, адже з'являється необхідність завантажувати всі підсторінки цільової веб-сторінки, тому була використана лише частина вибірки. Для того щоб суттєво скоротити часові витрати, можна спробувати обмежити кількість завантажених підсторінок, використовуючи бінарну вибірку - зібрати 6 випадкових посилань, дві з яких знаходяться на початку сторінки, наступні дві - в середині, а решта - в кінці. Для початку проведемо класифікацію за новою вибіркою з використанням тільки цільової веб-сторінки (точність 0.846). Виконаємо класифікацію за підсторінками (точність 0.690). Результат виявився набагато нижче. Можна спробувати об'єднати текст всіх підсторінок (точність 0.770). Даний підхід не дозволив досягти бажаного результату через обмеженість підсторінок, не всі підсторінки були доступні або відносилися до контенту цільової. Тому була використана та ж частина вибірки з бінарної категорією, але вже з усіма підсторінками. При класифікації за підсторінками результат виявився вищим: 0.851.

Даний метод дозволяє підвищити точність і використовувати новий вид атрибутів, але точність і швидкість отримання результату залежать від кількості підсторінок. Веб-сторінки з більш складним контентом для класифікації не дають позитивного результату через вміст змішаного набору категорій. Також цільові веб-сторінки, що містять зображення, не дозволяють отримати підсторінки, тому розглянемо можливість використання сусідніх веб-сторінок.

Висновок

В результаті досліджень було виявлено, що додавання адреси або заголовка у вигляді додаткових атрибутів не дозволило збільшити точність моделі. Можливо, для швидкої класифікації цей метод зручний, але точність настільки низька, що це не дозволяє використовувати його в даному випадку.

Використання Word2vec не принесло значних змін в результати. Можливо, якщо вибірка даних на навчання була б набагато більше, то точність могла б збільшитися завдяки наявності більшої кількості важливих слів в центроїді.

Описані методи, пов'язані з посиланнями, не дозволяють отримати очікуване поліпшення точності. Це пов'язано з правильним відбором

посилань. Важливо дослідити, які посилання найбільш корисні. Крім того, використання батьківських веб-сторінок можливо лише у вигляді дослідної роботи, коли в ідеальних вимогах можна зібрати за довгий період необхідні дані. У разі практичного застосування цей метод складно повторити через відсутність механізму швидкого отримання батьківських веб-сторінок. Також тривалість класифікації при використанні сусідніх веб-сторінок займає чимало часу через необхідність завантажувати їх. Підсторінки можуть бути отримані простим способом, але не дають високої точності, а лише дозволяють вирішити питання вибору категорії для сторінок з низьким рівнем впевненості.

Було розглянуто більшість відомих методів класифікації веб-сторінок та запропонована власна модель для класифікації веб-сторінок з точністю precision micro-averaging - 96%. Точність отриманої моделі не є ідеальною. Основну складність при класифікації веб-сторінок представляють ті сторінки, які не містять тексту. Можливо, додавання атрибутів такого типу може поліпшити якість класифікації, тому одним з можливих напрямків в дослідженні класифікації може бути використання нових видів атрибутів, таких, як зображення.

Література

1. Abdallah T. A., Iglesia B. URL-based web page classification-a new method for URL-based web page classification using n-gram language models // SCITEPRESS Digital Library-KDIR 2014-Intern. conf. on Knowledge Discovery and Information Retrieval. Rome, Italy, 2014.
2. Belmouhcine A., Benkhalifa M. Implicit Links based Web Page Representation for Web Page Classification // Proc. of the 5th Intern. conf. on Web Intelligence, Mining and Semantics. Larnaca, Cyprus, 2015.
3. Yanchang Zhao. R and Data Mining: Examples and Case Studies. Elsevier, 2012.

УДК 004.42

ДОСЛІДЖЕННЯ ОЗНАК ТЕКСТУ В ПРОЦЕСІ КЛАСИФІКАЦІЇ ДОКУМЕНТІВ

Кузьомка М. А, науковий керівник Боярінова Ю. Є

Сьогодні обсяг інформації, що доступна в Інтернеті, дуже великий і постійно зростає. Класифікація текстів носить міждисциплінарний характер, тому що зачіпає декілька галузей знань - обробку природної мови, машинне навчання, розпізнавання образів, статистику. Це пов'язано з тим, що численні фактори визначають ефективність роботи текстового класифікатора. На результати аналізу впливають попередня обробка даних, алгоритми машинного навчання, а також методи подання вихідної текстової інформації [1].

Причиною неоднорідності даних, що зібрані в Інтернеті, є різноманіття їх джерел: стрічки новин, повідомлення, огляди фільмів, рекламні оголошення тощо. Тому дані мають різні формати, часто істотно відрізняються їх стилі написання.

Документ може бути представлений двома різними способами. Для формалізації текстової інформації запропоновано використовувати просторову векторну модель, в якій документи відображаються як вектор ознак тексту. При цьому вихідні дані представляються як «мішок слів», в якому документ визначається у вигляді набору слів та їх частоти в документі. Таке представлення не залежить від послідовності слів в тексті. Векторна модель широко використовується в даний час для інформаційного пошуку і класифікації текстів.

Другий спосіб - представляти текст безпосередньо у вигляді рядків, в яких кожен документ визначається строгою послідовністю слів. Великі текстові класифікатори частіше використовують представлення «мішок слів» через свою простоту [2].

Найбільш поширена процедура, яка використовується в класифікаторах - це видалення «стоп-слів» (прийменників, сполучників тощо), які представляють собою загальні слова в документі і не є специфічними для різних класів. Також різні форми одного і того ж слова об'єднуються в одну ознаку. Наприклад, слова в однині та множині вважаються одним словом.

Одним з фундаментальних завдань при класифікації текстів є вибір характерних ознак. Важливість цієї процедури визначається великою розмірністю текстових об'єктів і наявністю нерелевантних ознак [3].

В якості ознак використовуються слова, частини слів, фрази або інші елементи тексту. Проблема класифікації полягає в тому, що тексти можуть містити велику кількість ознак, які придатні для ідентифікації. У цьому випадку застосовуються специфічні характеристики, наприклад, позначення часу або знаки пунктуації.

Зменшити кількість ознак можна шляхом об'єднання специфічних елементів, а також поданням стійких «штампів» з використанням спеціальних термінів. Можна також варіювати з різними форматами чисел, позначаючи всі числа як одна ознаку, або використовувати кілька ознак, застосовуючи, наприклад, чотири цифри дати (позначення року) в порівнянні з дійсними числами з десятковою крапкою.

Застосування певного методу зважування ознак має важливе значення для точності класифікації. Основні схеми зважування ознак текстів: бінарна («1» - ознака представлений в документі, «0» - немає), TF (Term Frequency) «частота терміна» - підрахунок частоти термінів, наявних в документі, DF (Document Frequency) «частота документів» - підрахунок частоти документів, що включають цей термін [1].

За винятком бінарного підходу, в якому поява ознак позначається як «1» і відсутність як «0», два інших підходу припускають знаходження ваг, засноване на частоті терміна або документа.

Хоча ці схеми часто використовуються як автономні, можливе застосування комбінованого математичного підходу. Частота документів і частота термінів, в основному, об'єднуються схемою зважування TF і зворотною DF, щоб використовувати іншу широко відому схему $TF\text{-}IDF$ (Term Frequency Inverse Document Frequency) «частота терміна - зворотна частота документу». Ідея цієї комбінації полягає в тому, що чим вище частота терміна в даному документі, тим більше він відображає його зміст. Крім того, чим більше документів, в яких зустрічається термін, тим менше ознак розрізняє зміст документів.

Деякі текстові класифікатори можуть також використовувати особливості розпізнавання «шапки» документів. Документи, як правило, мають додаткову структуру, метадані, пов'язані з кожним документом. Дані включають в себе поля: дата створення і формат документів, автора, назву статті.

При класифікації можна отримати кращий результат, привласнюючи збільшену вага різним зонам документів. Збільшення ваги слів назви є найбільш ефективним. Базуючись на припущенні, що більшу вагу в завданні класифікації мають слова першого речення або абзацу та, подвоївши ваги перерахованих зон, можна поліпшити якість класифікації.

Цією стратегією можна користуватися, збільшивши ваги слів в назві, або тільки в першому абзаці, або в абзаці, що має найбільшу кількість слів з назви чи ключових слів, або в першому і останньому абзаці.

Загалом позиційні методи визначення ознак і їх ваг, як правило, дають хороші результати за рахунок взаємної інформації та використання зонування документів. Вважається, що тільки диференціація ваг різних зон документа призводить до значного поліпшення ефективності класифікатора.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Барсегян А.А. Технологія аналізу даних: Data Mining, Visual Mining, Text Mining, OLAP / А.А. Барсегян, М.С. Купріянов, В.В. Степаненко. - СПб.: БХВ-Петербург, 2007.
2. Маннинг К.Д. Введення в інформаційний пошук. / К.Д. Маннінг, П. Рагхаван, Х. Шютце. – М.-СПб.-Киев: Вильямс, 2011.
3. Леонтьєва Н.Н. Автоматичне розуміння текстів. Системи, моделі, ресурси: навч. посібник / Н.Н. Леонтьєва. – М.: Academa, 2006.